

VMware vCenter Configuration Manager Troubleshooting Guide

VCM 5.3

This document supports the version of each product listed and supports all subsequent versions until the document is replaced by a new edition. To check for more recent editions of this document, see <http://www.vmware.com/support/pubs>.

EN-000476-00

vmware®

You can find the most up-to-date technical documentation on the VMware Web site at:

<http://www.vmware.com/support/>

The VMware Web site also provides the latest product updates.

If you have comments about this documentation, submit your feedback to:

docfeedback@vmware.com

© 2006-2010 VMware, Inc. All rights reserved. This product is protected by U.S. and international copyright and intellectual property laws. VMware products are covered by one or more patents listed at <http://www.vmware.com/go/patents>.

VMware is a registered trademark or trademark of VMware, Inc. in the United States and/or other jurisdictions. All other marks and names mentioned herein may be trademarks of their respective companies.

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

Contents

About This Book	5
General Troubleshooting Guidelines	7
Verify the Behavior is Negative	7
Isolate the Behavior	8
Identify External Factors	9
Check the Debug Log	10
Info Messages	11
Warning Messages	11
Error Messages	12
Exception Messages	12
Gather Information for VMware Customer Support	12
Types of Problems	13
User Interface	13
Security / Authentication	14
SQL Server	14
VCM Agent	15
UNIX Agent	15
Report Server	15
Internet Information Services (IIS)	16
Network Connectivity	17
Hardware and Performance Issues	17
How to Gather Diagnostic Files	19
Screenshots	19
Debug Logs	19
SQL Server Logs	20
IIS Logs	21
ARS Files	21
Syslog File (UNIX)	22
IE Tool Logs	22
Event Logs	22
System Information (msinfo32.exe)	23
ETL Logs (UNIX)	23
VCM Installation Logs	23
Patching Debug Information	23
Software Provisioning Troubleshooting	25
Troubleshooting Software Provisioning Repositories	25
When requesting a package from a repository, the package (.crate file) cannot be found	26
Repository.index is no longer valid	27
Crates.gz file is corrupt	27
Too many users adding new platforms and sections	27
Repository not found when reindexing or publishing packages to a repository	28
Troubleshooting Package Studio	28
Error when starting Package Studio: "Cannot create instance of 'RepositoryEditorViewModel' defined in assembly 'PackageStudio...'"	28
The Generate button is grayed out	29

Error when adding a Provides: the dependency package name is invalid	30
When a package is created, signed, saved as project, and then generated, the package is no longer signed	31
The installed size of the application is set to nnn, but systems with nnn run out of space during installation	31
The package installs the application but does not uninstall	31
Files are added to the Project Data Directory, but they are not displayed in the list	31
I am uncertain if the package requires a reboot to install or I only want the installation to reboot sometimes	32
Gathering Software Provisioning Logs on the Collector	32
Gathering Software Provisioning Logs on the Agent	32
Windows Agent Installation	35
Pre-Install Environment	35
Network	35
Collector	35
Manual Agent Install	44
Protocol Specific	45
UNIX Agent Troubleshooting	47
Agent Directory Structure	47
Collector Certificates	56
Patch Assessment	57
Directories Created During an Inspection	58
Saving Executed Scripts and Results	59
UNIX Agent Error Scenarios	60
If the installation reports an error	60
The Collector cannot ping the Agent	60
Agent fails to return data	61
Monitoring network traffic	63
The Collector reports the job succeeded, but there is still no data	66
Index	67

About This Book

This manual, *Troubleshooting Guide for VMware vCenter Configuration Manager*, explains the types of problems that may occur with VMware vCenter Configuration Manager, the diagnostic material VMware Customer Support will need to analyze the problem, and the steps to retrieve that information.

Intended Audience

The information presented in this manual is written for system administrators who are experienced Windows or UNIX/Linux/Mac OS X system administrators and who are familiar with managing network users and resources, and performing system maintenance.

To use the information in this guide effectively, you must have a basic understanding of how to configure network resources, install software, and administer operating systems. You also need to fully understand your network's topology and resource naming conventions.

Document Feedback

VMware welcomes your suggestions for improving our documentation. If you have comments, send your feedback to docfeedback@vmware.com.

VMware VCM Documentation

The vCenter Configuration Manager (VCM) documentation consists of the VCM documentation set.

Technical Support and Education Resources

The following technical support resources are available to you. To access the current version of this book and other books, go to <http://www.vmware.com/support/pubs>.

Online and Telephone Support

To use online support to submit technical support requests, view your product and contract information, and register your products, go to <http://www.vmware.com/support>.

Customers with appropriate support contracts should use telephone support for priority 1 issues. Go to http://www.vmware.com/support/phone_support.html.

Support Offerings

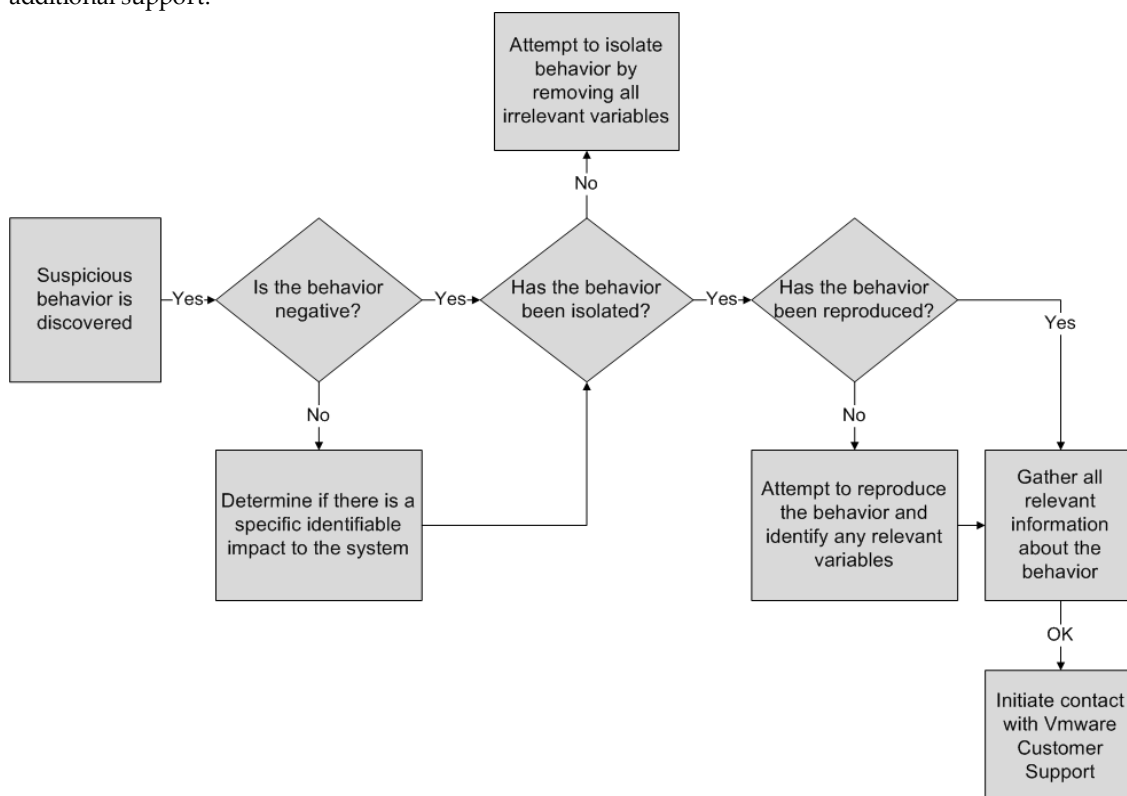
To find out how VMware support offerings can help meet your business needs, go to <http://www.vmware.com/support/services>.

VMware Professional Services

VMware Education Services courses offer extensive hands-on labs, case study examples, and course materials designed to be used as on-the-job reference tools. Courses are available onsite, in the classroom, and live online. For onsite pilot programs and implementation best practices, VMware Consulting Services provides offerings to help you assess, plan, build, and manage your virtual environment. To access information about education classes, certification programs, and consulting services, go to <http://www.vmware.com/services>.

General Troubleshooting Guidelines

This document attempts to describe some of the basic steps you can take before contacting VMware Customer Support. Below is a simple flowchart that provides troubleshooting procedures, allowing you to resolve simple issues on your own. Then, if necessary, you can contact VMware Customer Support for additional support.



Verify the Behavior is Negative

There are a few messages that appear in VCM and VCM Patching that may be interpreted as errors, when in fact they are merely warnings. Other behaviors may be operating exactly as defined, but since the result is not what is expected, it could be interpreted as wrong behavior.

For example, when creating a Compliance rule, it is important to realize that the conditional statements must be set up to highlight systems that fail to meet the described normal condition as non-compliant. The VCM User Interface highlights this fact in the Compliance rule creation wizard on the condition definition page displayed below:



This construction may be confusing. A user might intuitively create a conditional statement that describes the abnormal condition they want to eliminate. As a result, they find that VCM returns a list of nearly all their machines and flags them as non-compliant with the abnormal rule. In reality, nearly all these systems might be compliant and VCM has merely performed exactly as requested.

The first step to troubleshooting VCM, therefore, is to first verify that the suspect behavior is a negative behavior or one that is not within the normal operating parameters of VCM. The VCM Help file is the perfect tool to use when trying to determine normal behavior. There is a Help button on almost every screen of each VCM wizard as well as in the upper-right corner of the Portal. The Help buttons, on the Portal or in a wizard, display context-sensitive help topics relevant to the area of VCM in which you are currently working.

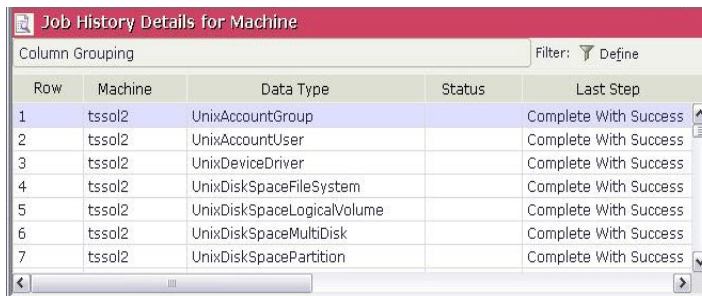
Validate that the message or behavior seen is not described in the online Help before assuming the behavior is negative or undesirable.

Isolate the Behavior

The next step to take when troubleshooting an undesired behavior is to try to isolate it. This is done by eliminating, one by one, all factors that may be contributing to the behavior until the bare minimum number of factors are in play when the error occurs.

For example, if you encounter some problem on a single machine during a collection for 50 machines and 10 data types per machine, the first step would be to eliminate the 49 machines that did not exhibit the same behavior, and run a collection for the same 10 data types against the single problem machine. You can usually isolate which machine failed by viewing the details of the job in the Job History screen on the Administration slider. If you witness the same behavior, you then eliminate data types (one or two at a time) until you find the data type that is causing the behavior. Again, the job details on the Job History screen (as seen below) may give you some indication as to which data types failed.





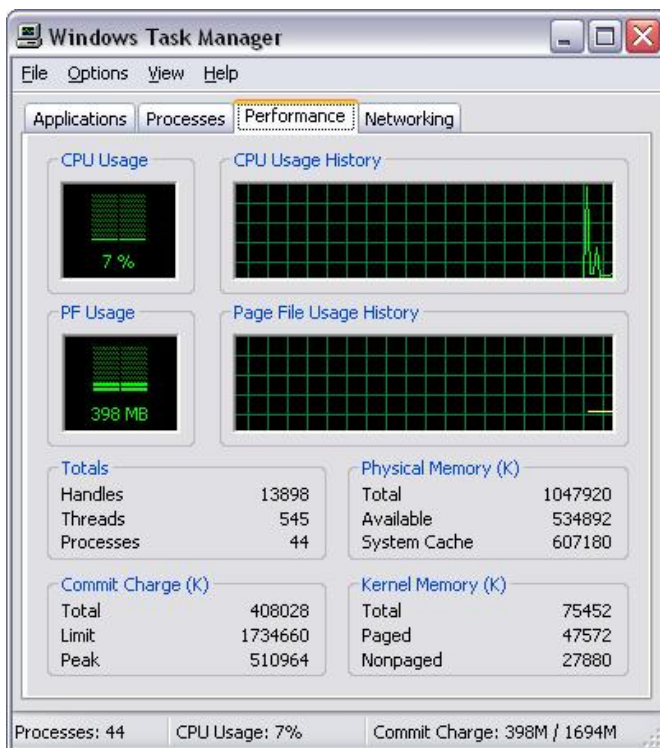
Row	Machine	Data Type	Status	Last Step
1	tssol2	UnixAccountGroup		Complete With Success
2	tssol2	UnixAccountUser		Complete With Success
3	tssol2	UnixDeviceDriver		Complete With Success
4	tssol2	UnixDiskSpaceFileSystem		Complete With Success
5	tssol2	UnixDiskSpaceLogicalVolume		Complete With Success
6	tssol2	UnixDiskSpaceMultiDisk		Complete With Success
7	tssol2	UnixDiskSpacePartition		Complete With Success

The goal for this trial-and-error elimination process is to understand when the undesired behavior occurs so you can reproduce the behavior consistently and at will. Once you know exactly how to reproduce the error at any time, be sure to document those steps for future investigation.

Identify External Factors

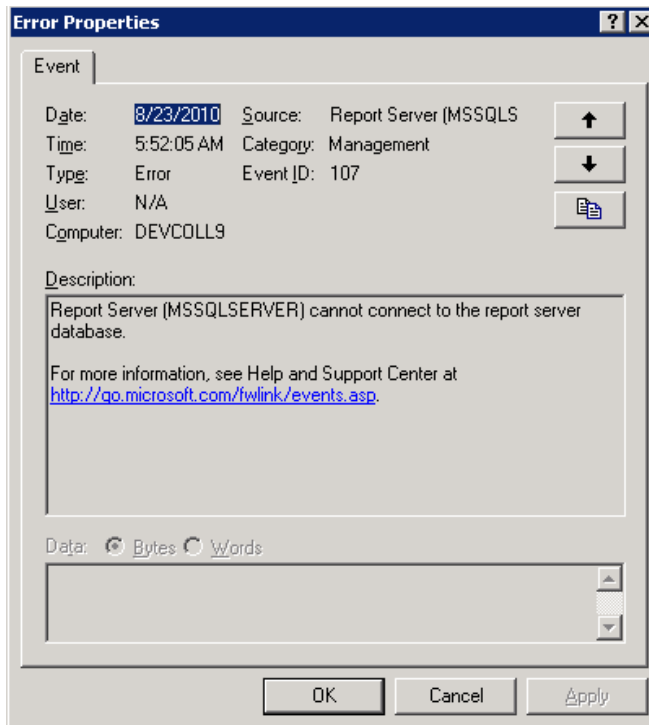
Sometimes the behavior is seemingly unpredictable and the above method will not succeed in isolating it. For these types of issues, it is best to take special note of the status of the environment when the behavior appears. Due to the amount of data processed by VCM, the systems running the database and the main application are subject to periods of high resource consumption.

System load can be a good troubleshooting indicator when problems arise. When investigating an undesired behavior, take note of the system load, memory usage, network traffic, time of day, other running applications, or anything else that may indicate a correlation with the behavior on both the Collector machine and the Agent machine. For Windows machines, Task Manager should provide you with this information (as seen in the image below).



For example, if collections are stalling at a specific step in the process, you may occasionally find that a single CPU is running at 100% capacity. This may indicate any number of problems, but knowing that the CPU running at 100% always correlates with the behavior you are investigating is the most important thing at this stage. With this approach, you may be able to isolate a specific time of day, or a set of conditions that must be present for the undesired behavior to occur.

Another good method for identifying external factors is reviewing Event Log. Many simple problems may be detected by reviewing the Security and Application logs. The most common errors you will see in the Event Log are authentication problems (as seen below), but any error or warning message occurring in the Event Viewer during the time frame surrounding the undesired behavior is suspect, especially if the same error message always accompanies the undesired behavior.



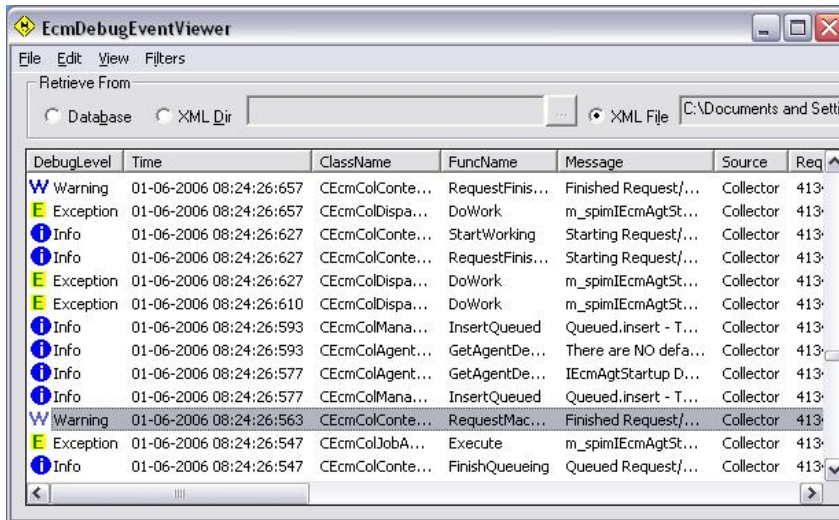
Errors and behaviors that involve hardware problems or failures may be extremely erratic. You may see a series of seemingly unrelated errors occur in sequence, or you may see the same error at random times. You may also see that the behavior of VCM appears to degrade over time. These sorts of behaviors may indicate a need for system hardware diagnostics.

Check the Debug Log

VCM produces several files in various locations through its normal operation with the extension “.dbe”. The .dbe files contain information about VCM operations. The Collector also writes debug information to the database whenever jobs are running in the Portal.

The Collector log is the most important log file as it records the primary functionality of VCM, and when VMware Customer Support technicians speak of “the debug log” in the singular, they are typically referring to this Collector debug log. Debug logs are a common troubleshooting tool for VMware Customer Support technicians, so you may already be very familiar with the process of gathering them. You may even recognize several of the messages mentioned in the logs, which may or may not be helpful.

It is important to remember that the debug logs produced by VCM were never designed for the average user. The messages contained in the logs have been specifically created to help developers understand why VCM is operating the way it is. Always review the debug logs yourself and when necessary, forward the log in its entirety on to VMware Customer Support for analysis. A sample debug log is provided below.



It is not necessary for any support technician to be familiar with every message produced in a debug log, but it is extremely helpful to know what the different types of errors logged in the debug log can mean when determining the cause for a specific behavior. There are four major categories of error messages:

- Info messages
- Warning messages
- Error messages
- Exception messages

Info Messages

Info messages are the most common message category that you will find if VCM is configured to save them. By default, Info messages are discarded by VCM when writing to the debug logs because the volume of information produced by VCM under normal circumstances will make these log files grow substantially. If, however, Info messages are enabled under the Administration slider, nearly every action VCM performs will be logged in the various debug logs. Info messages are critical when diagnosing an undesired behavior because it gives the developers a context surrounding the behavior.

Warning Messages

Warning messages are the next most common message category you will find in debug logs. Warnings indicate a point in the process where an unexpected situation arose, but do not necessarily identify a problem with VCM, as many common situations such as network timeouts or authentication problems will produce a warning. Also, warning messages should not stop VCM from executing normal processes, but they are intended to alert the developers of a possible problem.

Error Messages

Error messages indicate a more serious problem. VCM processes will not halt because of an error, but they may not be able to continue normally. For example, if the collection of a specific data type produces a value that is not of the same type the database is expecting, VCM may have to discard the collected information for that entire data type. This would produce an error message to highlight the fact that something did not work properly during the latest process. In this case, VCM was able to recover and continue, but perhaps not with the desired result.

Exception Messages

Exceptions messages indicate a fatal problem that VCM cannot handle. These types of problems will likely cause a complete failure of the current process. One common exception occurs with overseas customers. When a collector machine is set up to use a regional date/time notation other than English (United States), which uses a date/time notation of mm/dd/yyyy to represent the date, collections to UNIX machines will usually fail. Many European date/time notations use dd/mm/yyyy to represent the date. The message reported in the debug log is an exception with the message “Arithmetic overflow converting expression to a data type datetime.” In essence, this message states that VCM could not convert the UNIX date/time notation to the date/time notation used by the SQL database. With this exception occurring, complete UNIX collections will fail.

Gather Information for VMware Customer Support

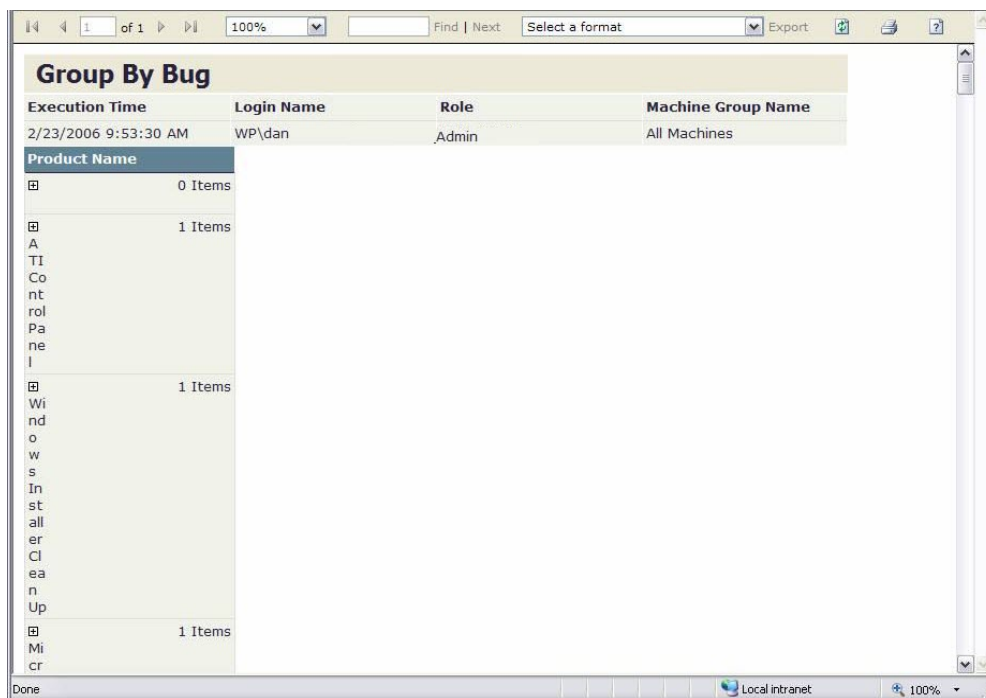
When reporting an issue to VMware Customer Support, you will save a great deal of time if you are prepared with as much information about the problem as possible. Use the “Types of Problems” section of this document to determine which diagnostic files VMware Customer Support will need in order to analyze the behavior.

Types of Problems

Most VCM problems will fall into one of several categories that may make them easier to identify. Each category is representative of one of the areas critical for proper VCM functionality. When troubleshooting a behavior, keep these categories in mind as they may add to your understanding of how VCM operates as a whole.

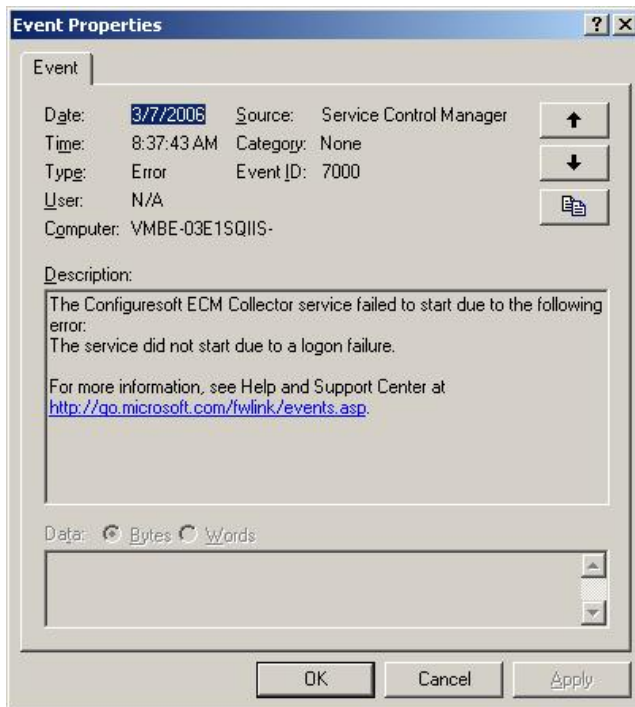
User Interface

The User Interface (UI) is the main way users interact with VCM. The UI communicates primarily with the VCM database to retrieve and submit information. While many unwanted behaviors may reveal themselves in the UI, the UI is rarely the root cause. The most common problems directly associated with the UI are display anomalies. Duplicate columns in the Data Grid, inappropriately disabled text fields, and data formatting problems are all examples of UI-related problems. Screenshots typically provide the most helpful information when relating these problems to VMware Customer Support. Below is an example of a UI problem where the headers of a report are compressed into only a two or three character width.



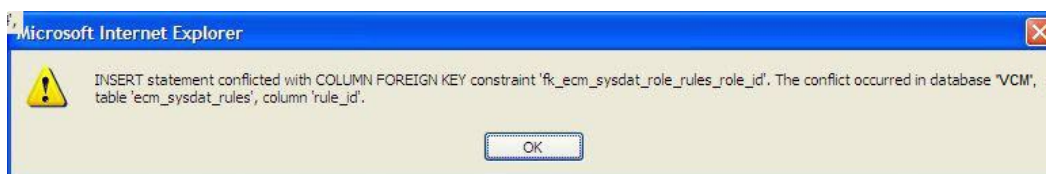
Security / Authentication

One of VCM's primary roles is multi-system administration. Therefore, VCM must be granted authority to act as an administrator on all the machines under its control. Similarly, the number of places where that authority may be removed or truncated is equally large. Common authentication problems may include changing passwords, dropped administration access, and implementation of additional security measures (such as a proxy server) without updating VCM. System access is only half of the access required by VCM. VCM also requires access to the SQL Server database, with the authority to insert, modify, and delete data within it. Access problems of both sorts may reveal themselves in the UI, in the Event Viewer, as well as in the debug log. Screenshots of UI errors combined with debug logs are the best information to send VMware Customer Support when reporting problems. Below is an example of an authentication error found in the Event Viewer.



SQL Server

Nearly all data associated with VCM is stored in the SQL database. The VCM database contains the collected system data from the VCM Agents. The VCM_Coll database contains information about the UI, such as Collector settings and options. The VCM_Raw database is a performance-enhancing database used to temporarily store information from collections before bulk insertion into the VCM database. Finally, the VCM_UNIX database contains collected information from any UNIX Agents in the customer's environment. SQL Server errors may include resource problems, disk space problems, or authentication problems, among others. Improper tuning of a SQL Server also may cause performance problems such as data bottlenecks. Evidence of SQL Server errors may appear in the UI or in the debug log, but messages directly relating to the SQL Server should appear in the SQL Server logs. If you suspect a SQL Server error, the SQL Server logs, the system and application Event Logs, and the VCM debug log are the most helpful sources of information for VMware Customer Support. Below is one example of an error you may see associated with SQL Server.

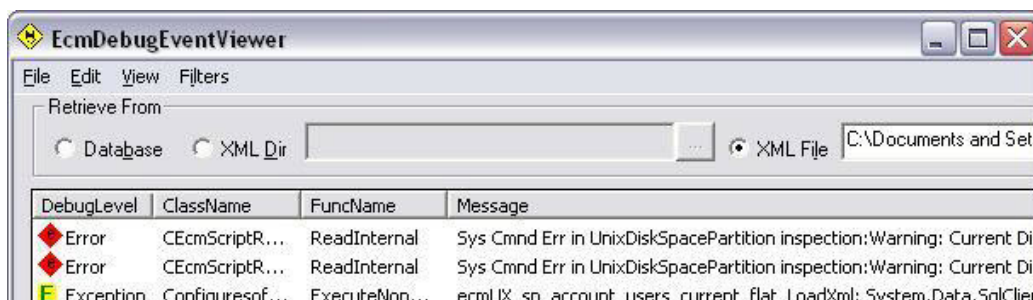


VCM Agent

The VCM Agents are the mechanism by which VCM collects information from managed machines. If the Agent is not functioning properly, collections from that machine may fail. The Running Jobs UI is the first component to identify problems with an Agent. A typical error message for an Agent that is unable to start collecting might be “There was a problem parsing the agent instructions document.” This message indicates there was some conflict in the instruction set sent to the Agent that it could not process, and failed as a result. When troubleshooting Agent errors, a debug log and any available Agent ARS files is the best information to send to VMware Customer Support when requesting assistance. Also, check the Event Log for any application or system messages that may indicate why the Agent failed.

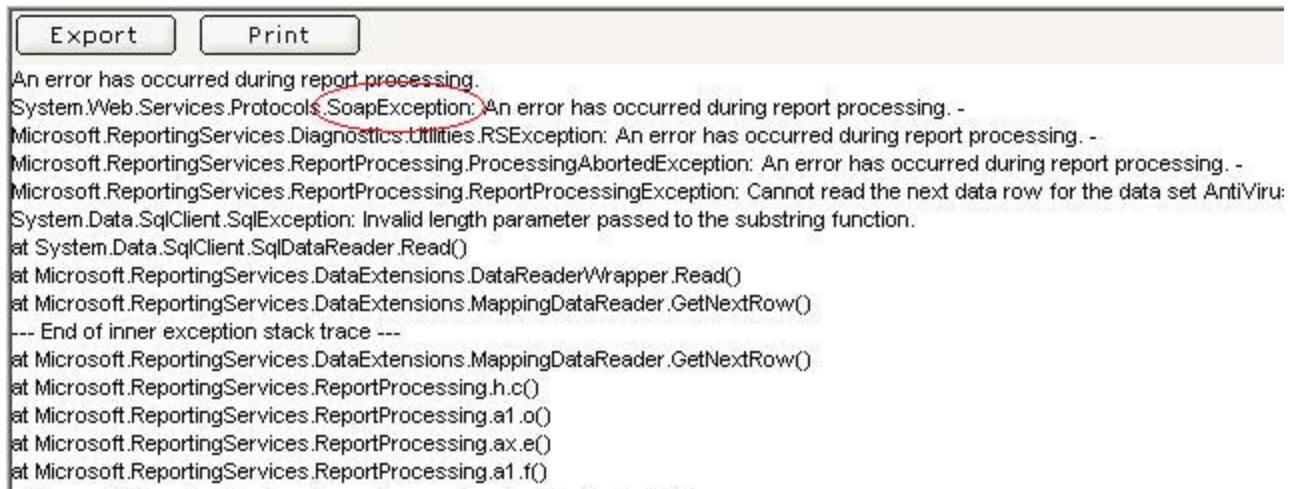
UNIX Agent

UNIX operates very differently from Windows. To accommodate the differences, VCM uses a separate Agent for UNIX/Linux-based platforms. Troubleshooting UNIX Agent problems can be a little bit easier than the standard Windows Agent because many operations available for Windows Agents are not yet available on the UNIX side. Compliance enforcement is not yet available for the UNIX Agent, and therefore should be eliminated from your troubleshooting process. For UNIX Agent errors a debug log, UNIX Agent debug log, UNIX Agent ZRP files, and a system log are all helpful information to VMware Customer Support technicians when troubleshooting UNIX Agent issues. Below is a typical UNIX error shown in the debug log.



Report Server

The Report Server is responsible for the graphical displays of information in VCM and for the VCM scheduled reports. Report error will normally appear in the UI.



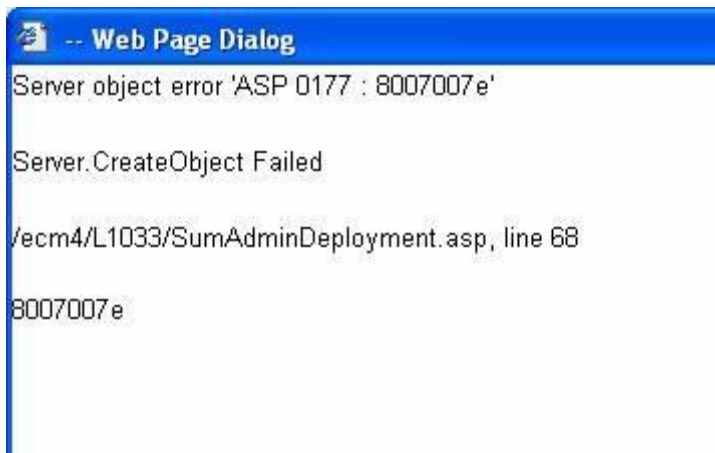
You can easily determine if there is a problem with the Report Server specifically by navigating to the Report Server homepage to see if the native interface for the Report Server is operational. This address is usually just the name of the Report Server machine followed by “/Reports”. See the following example:

<http://localhost/Reports>

If the same type of problem you discovered in the UI also occurs on this Reports, then the problem is almost certainly with the Report Server itself, and not with VCM. If the Report Server interface appears without error, the cause is likely in VCM. For these errors, screenshots, any support-requested SQL profiler trace files, and a debug log are the files the VMware Customer Support technicians should see.

Internet Information Services (IIS)

The VCM UI is displayed using Microsoft’s Internet Information Server (IIS). Occasionally you may see error messages in the UI that you typically see when failing to connect to a webpage. Error messages like “404 File Not Found” or “403 Forbidden” are typical IIS messages, and relate to the improper functioning of the IIS configuration in relation to VCM or Report Server pages. You may also see ASP.NET errors when IIS is involved as seen below.



When these errors occur, please send screenshots and any relevant entries in the IIS logs when requesting assistance from VMware Customer Support.

Network Connectivity

VCM relies heavily on network connectivity to all Agents. In systems where you have a split installation, failed network connectivity can also cause problems with VCM. Failures in network connectivity are not likely to occur across all systems involved with the operation of VCM at the same time, so issues with the network are occasionally easier to diagnose. Issues with the network are usually first discovered in the UI with the failure of some VCM or VCM Patching job to a single or subset of machines. A common error message associated with connectivity problems is “Ping Failed” as seen below.



Row	Machine	
1	testmachine3	Succeeded
2	testmachine1	Create connection of 6: PingFailed
3	testmachine2	Create connection of 6: PingFailed
4	testmachine5	INSERT statement conflicted with COLUMN

The simplest way to test network connectivity is to ping the Agent from the Collector. If the environment contains a firewall, if the Agent is a UNIX agent, or if the Agent machine is using the HTTP protocol instead of the DCOM protocol for its primary communication method, it may be necessary to connect to the HTTP port of the Agent on the target machine from the Collector. This can be done by using the telnet command in the following way:

```
C:\> telnet target_machine 26542
```

In the above example, the name of the machine we are testing is “target_machine” and the HTTP port used is 26542. A successful test of this type should return nothing but a blank screen. This is the target system answering the connection request and awaiting further instructions. You can break this connection by typing “Ctrl-]” and then typing “quit” to exit the telnet program. If you receive any other message, such as “Connection refused” or “Connection timeout”, that is a good indication of network communication problems. These issues will need to be worked out with the local network support individuals before VCM troubleshooting may continue.

Hardware and Performance Issues

Hardware and performance issues are some of the most difficult issues to diagnose because they are usually sporadic and random. In fact, there may be times where the only constant you can count on is random errors. This is the time to look at possible hardware and/or performance problems. Running out of disk space is the most common hardware problem. Simply check the file manager to verify that enough disk space is present for both the database and for VCM itself. Next, make sure that there is enough memory and enough CPU cycles to ensure that the VCM services and processes can function properly. If VCM is competing with several other processes on the system that are unrelated to VCM, performance may degrade to the point that errors and exceptions are produced in the debug log. Finally, it may be necessary to run diagnostics on the hardware components, such as the memory chips, processor(s) and system boards. Refer to VCM documentation to see the recommended hardware configurations needed to run VCM properly in the different possible environments.

How to Gather Diagnostic Files

You may need to review or gather one or more of the following files to use for diagnostic purposes. The following sections provide the steps needed to locate the files.

Screenshots

Screenshots are one of the easiest ways to demonstrate the exact behavior you see on your system. A single screenshot can show an error message, and a series of screenshots can show a change in behavior over time. A screenshot series is also good for verifying the values you selected in a wizard screen-by-screen.

To capture a full desktop screenshot:

1. Click the Print Screen button (PrtScr) on the keyboard.
2. Open an new email message or a Word document.
3. Press Ctrl-v to paste the image into the message or document.

To capture a screenshot of a single window:

1. Make certain the window you want to capture is active.
2. Hold down the Alt key and press the Print Screen key (PrtScr).
3. Open a new email message or Word document.
4. Press Ctrl-v to paste the image into the message or document.

Debug Logs

Debug logs are the primary way for our developers to trace the behavior of VCM. The most important debug log is the Collector debug log, and must be extracted from the database using the EcmDebugEventViewer.exe file located in <drive>:\Program Files\VMware\VCM\Tools. The following procedure is used to extract the Collector debug log complete with INFO messages:

1. Start the VCM portal.
2. Select **Administration | Settings | General Settings | Collector**. The **Collector Settings** data grid appears.
3. Select the **Type of information that should be logged**, and then click **Edit Settings**. The **Edit Settings** page appears.
4. Select all the options: Exception, Error, Warning, and Info.
5. Complete the other pages in the wizard, and then click **Finish**.

6. Select **Administration | Settings | General Settings | Database**. The **Database** data grid appears.
- 7.
8. Select the **Type of information that should be logged**, and then click **Edit Settings**. The **Edit Settings** page appears.
9. Select all the options: **Exception, Error, Warning, and Info**.
10. Complete the other pages in the wizard, and then click **Finish**.
11. Select **Administration | Settings | Windows | Agent - General**. The **Agent General Settings** data grid appears.
12. Select the **Type of information that should be logged**, and then click **Edit Settings**. The **Edit Settings** page appears.
13. Select all the options: **Exception, Error, Warning, and Info**.
14. Complete the other pages in the wizard, and then click **Finish**.
15. Select **Administration | Settings | UNIX | Agent - General**. The **Agent General Settings** data grid appears.
16. Select the **Type of information that should be logged**, and then click **Edit Settings**. The **Edit Settings** page appears.
17. Select all the options: **Exception, Error, Warning, and Info**.
18. Complete the other pages in the wizard, and then click **Finish**.
19. Restart the VCM Collector service in the Services manager.
20. Now run the job in question again. You must wait 5 minutes after it completes before proceeding.
21. Minimize the console and navigate to <Drive>:\Program Files\VMware\VCM\Tools. In this directory, double-click `ECMDebugEventViewer.exe`.
22. In **ECMDebugEventViewer**, click **Filter Settings**. The **Filter Settings** dialog box appears.
23. Select all the check boxes in the **Message Type** and **Message Source** areas. Click **OK**.
24. In the **Data Source** area, type the names of the servers and databases.
25. Click **OK** to close the dialog box.
26. Click **Date/Time**. The **Data/Time** dialog box appears.
27. Select the **between** option, specify the dates and add five or more minutes to the beginning and end times during which the collection ran.
28. Click **File**, and then select **Fetch**. The displayed data is refreshed.
29. Click **File**, and then select **Fetch Next**. Continue the **Fetch Next** process until no additional data is added to the displayed debug log.
30. Click **File** and select **Save as Dbe**. Save the file as a DBE file and note where it was saved for later use.
31. Repeat steps 1-4 and reverse the settings for the Collector, Database, and Windows Agent and UNIX Agent objects back to their original logging levels (usually **Exception, Error, and Warning**).

SQL Server Logs

The SQL Server logs keep track of events related to the database operation. Use the following process to extract them from SQL Server:

1. Log on to the machine with the SQL Server used by VCM.
2. Select **Start | All Programs | Microsoft SQL Server | SQL Server Management Studio**.
3. Select the **Server name** and **Authentication** method, as needed, and then click **Connect**.
4. In the **Object Explorer** pane, expand <ServerName> | **Management | SQL Server Logs**.
5. To export, right-click the log name, and then select **View SQL Server Log**. The **Log File Viewer** displays the logs.
6. Click **Export**. The **Export Last Fully Retrieved Log** dialog box appears.
7. Save each file out with an appropriate name and note where they are saved. You will need this information later.
8. Repeat the above steps for each log you are exporting.
9. Close SQL Server Management Studio and collect the saved log files.

IIS Logs

The IIS logs keep track of events related to the web server operation. These must be extracted from the IIS Manager. Use the following process:

1. Log on to the Collector machine.
2. Click **Start | Administrative Tools | Internet Information Services (IIS) Manager**. The **Internet Information Services (IIS) Manager** window appears.
3. Expand **Internet Information Services | <MachineName> | Web Sites**.
4. Right-click **Default Web Site**, and then select **Properties**. The **Default Web Site Properties** dialog box appears.
5. Verify that the **Enable Logging** check box is selected.
6. In the **Active log format** drop-down list, select **W3C Extended Log File Format**, and then click **Properties**. The **Logging Properties** dialog box appears.
7. By default they are stored in C:\WINDOWS\system32\LogFiles\W3SVC1\“. Each log file will have the notation “xyymmdd.log” where yymmdd represents the date.
8. Collect the relevant files.

ARS Files

The ARS files are the raw files sent to the Collector by the Agent. These files are typically deleted after the Collector processes the data contained in them and inserts that data into the database. You can force the Collector to retain that information, which can be a vital troubleshooting tool for the developers. Here is the process for collecting them:

1. Log on to the Collector.
2. Click **Start | Run**, and then type `regedit` in the text box.
3. Expand `\HKEY_LOCAL_MACHINES\Software\Configuresoft\ECM\4.0\Agent`. If that key does not exist, right-click on the 4.0 key and select **New | Key**. Name the key `Agent`.
4. Select the Agent key named **AreResultsSaved**. Verify the value is 1. If the value does not exist, create a new DWORD value named `AreResultsSaved` and set it to 1.

5. Expand \HKEY_LOCAL_MACHINES\Software\Configuresoft\ECM\4.0\Collector. Verify the value is 1. If that key does not exist, right-click on the 4.0 key and select **New | Key**. Name the key Collector.
6. Look for a value within the Collector key called **AreResultsSaved**. If the value does not exist, create a new DWORD value named **AreResultsSaved** and set it to 1.
7. Stop all running jobs (or make sure that no jobs are currently running).
8. Navigate to <Drive>:\Program Files\VMware\VCM\CollectorData.
9. When there are not jobs running, delete any existing folders in this directory. If jobs are running, wait until they are completed before continuing.
10. Start a collection. Make note of the Job ID in the Running Jobs window.
11. Once the job is completed in the Job Manager, look for a CollectorData folder with the same name as the Job ID of the just-completed job.
12. Compress the entire folder using a utility such as WinZip.
13. Open regedit again and set the Agent and Collector **AreResultsSaved** values to 0.

Syslog File (UNIX)

The syslog file in UNIX is similar to the Event Log in Windows as it records a great deal of system information. The location of this file may vary from system to system, but the following process should help you find it and copy it.

1. Log into the Agent system as the 'root' user.
2. Use 'cat' to view the **/etc/syslog.conf** file. Look for where the 'messages' file is located.
3. Collect a copy of the messages file and transfer the copy to your local workstation or Collector machine.

IE Tool Logs

The Import Export or IE tool used with VCM creates its own debug file when problems arise. These debug files may indicate where failures are occurring during the export of information from VCM or the import of information to VCM.

1. Log on to the machine with the IE tool installed.
2. Navigate to <Drive>:\Documents and Settings\All Users\Application Data\Configuresoft\ECMImportExport.
3. Gather any ".dbe" files contained in this directory.

Event Logs

Event logs are the primary location to find errors thrown by any operation on a machine that Windows detects. The Event logs are divided into 3 categories: Application, Security, and System. VCM errors will almost always fall under either Application or System. Use the Event Viewer to view these logs and to export them for review by VMware Customer Support.

1. Log on to the Collector machine
2. Select **Start | Administrative Tools | Event Viewer**. The **Event Viewer** appears.
3. Expand **Event Viewer** in the left pane.
4. Select an Event category, such as Application or System.
5. Click the Action menu and select **Save Log File As**. Type an appropriate name and click **Save**.

System Information (msinfo32.exe)

Occasionally, VMware Customer Support may ask for an msinfo32.exe file. Msinfo32.exe is an application that provides a fairly accurate and detailed snapshot of a system's current activity.

1. Log on to the Collector machine
2. Click **Start | Run** and type `msinfo32.exe` in the text box.
3. Click **File**. Type an appropriate name and click **Save**. The save process may take a few minutes.

ETL Logs (UNIX)

ETL logs are UNIX logs. These logs document the more detailed workings of the ETL service.

1. Log into the Collector machine.
2. Navigate to `<drive>:\Documents and Settings\All Users\Application Data\Configuresoft\ECM\ExceptionLog`.
3. Gather all the logs beginning with "etl".

VCM Installation Logs

When installing VCM, the installer script will place log files documenting the progress and status of the VCM installation process.

1. Log into the Collector machine as the user used to install VCM.
2. Click **Start | Run**. Type `%TEMP%` in the text box. You should be in the `/temp` directory.
3. Collect all files in `_csi_installation` directory in a zip file.

Patching Debug Information

The VCM Patching module creates its own logging information for Windows as a supplement to the VCM debug file. VCM Patching logging must first be turned on, however, by setting a switch in the VCM database.

1. Open **SQL Server Management Studio** either on the Collector machine, or a local workstation.
2. Connect to the VCM database using your preferred authentication method.
3. Click **New Query**, located on the toolbar. A blank execution pane appears.
4. Select the VCM database in the drop-down list, located on the toolbar.
5. Run the following query to turn VCM Patching debugging on:

```
update csi_hf_settings set val = '1' where setting = 'debug'
```
6. Select **Start | Administrative Tools | Services**. The Services window appears.
7. Restart the **VCM Patch Management** service.
8. Execute the VCM Patching process that produces the undesired behavior.

9. On the collector navigate to <drive>:\Program Files\VMware\VCM\SUM\Collector and look for the following files:
 - CSISUMWorker_SumDBDebug.txt
 - CSISUMSvc_SumDBDebug.txt
 - CSISUMSvc_debug.txt
10. Gather all these “.txt” files together.

Software Provisioning Troubleshooting

When troubleshooting software provisioning, there are three main components: the software repository, the Package Studio, and the Package Manager (the commands to the Package Manager are issued from VCM. The troubleshooting information provided here will help you identify and resolve problems, as well as provide you with a method for gathering the appropriate files required to troubleshoot problems for which a remediation is not provided.

Troubleshooting Software Provisioning Repositories

The following are problems you may encounter when working with software provisioning software repositories. Possible solutions are also provided.

When requesting a package from a repository, the package (.crate file) cannot be found

Possible Cause	Remediation
The package (.crate file) was deleted from the repository.	<ol style="list-style-type: none"> 1. Remove the package entries from the <code><path>\VMware\Tools\Repository\hive\repository.index</code> file. 2. Reindex all platforms and sections from which you removed the entry. To reindex, run <code>C:\Repository>"C:\Program Files\VMware\VCM\Tools\Package Studio\hive.exe" reindex "C:\Repository\dists\Any\Release\binary-<platform>" <platform> <section></code>. 3. Using Package Studio to republish the package to the repository.
The package was copied to the repository rather than published.	Publish the package to the repository using the Package Studio. Do not use the version of the package copied to the repository. You can either publish the locally saved version of the package to the selected repository platform and section, or you can copy the repository version to another location on the machine, and then publish it to the repository using the Package Studio.
The package was copied to the repository and the repository.index file was updated, but the platform and section were not reindexed and the package was not added to the crates.gz file.	Reindex all platforms and sections from which you removed the entry. To reindex, run <code>C:\Repository>"C:\Program Files\VMware\VCM\Tools\Package Studio\hive.exe" reindex "C:\Repository\dists\Any\Release\binary-<platform>" <platform> <section></code> .
The repository was manually reorganized.	<ol style="list-style-type: none"> 1. Update the Repository.index file to remove the old locations and add the new locations. 2. Reindex all platforms and sections from which you removed the entry. To reindex, run <code>C:\Repository>"C:\Program Files\VMware\VCM\Tools\Package Studio\hive.exe" reindex "C:\Repository\dists\Any\Release\binary-<platform>" <platform> <section></code>.
Package was published to the repository using the hive.exe command or Package Studio, but the repository entry was not included in the Repository.xml file.	Add the repository entry to the Repository.xml using Package Manager. Add to the same Platform and Section to which the package was published.

Repository.index is no longer valid

Possible Cause	Remediation
The Repository.index was manually edited and the xml code is corrupted.	Review the xml code for missing tags or other simple problems.
	<p>If fixing missing tags does not solve the error:</p> <ol style="list-style-type: none"> 1. Create a .bat file with publish commands for all the packages (.crate files), platforms, and sections in the Repository.index xml file. 2. Rename the failing Repository.index. 3. Create a new Repository.index file with an empty <RepositoryIndex/> tag. 4. Run the .bat file. This action should recreate the index in the Repository.index file and refresh all the crates.gz files.

Crates.gz file is corrupt

Possible Cause	Remediation
The file was manually edited.	Reindex all platforms and sections from which you removed the entry. To reindex, run <code>C:\Repository>"C:\Program Files\VMware\VC\Tools\Package Studio\hive.exe" reindex "C:\Repository\dists\Any\Release\binary-<platform>" <platform> <section>.</code>

Too many users adding new platforms and sections

Possible Cause	Remediation
All users have permission to add platforms and sections to the repository.toc.	Disable the write permission to the repository.toc file for users you do not want adding platforms and sections.

Repository not found when reindexing or publishing packages to a repository

Possible Cause	Remediation
Incorrect path to the repository.	When reindexing or publishing, use the fully qualified path to the root of the repository. For example, when publishing a package: C:\Repository>"C:\Program Files\VMware\VCM\Tools\Package Studio\hive.exe" publish "c:\Repository\internetexplorer_8.0_x86.crate" crates\i <platform> {<section>}.

Troubleshooting Package Studio

The following are problems you may encounter when working with the Package Studio. Possible solutions are provided.

Error when starting Package Studio: “Cannot create instance of ‘RepositoryEditorViewModel’ defined in assembly ‘PackageStudio...’”

Possible Cause	Remediation
The Package Studio was installed before the repository was installed. The installation process was unable to populate the repository location, and if the default repository directory does not exist, the error appears.	<ol style="list-style-type: none"> If it is not installed, install the repository,. Open the PackageStudio.exe.config file with a text editing application. The default location is C:\Program Files\VMware\VCM\Tools\Package Studio. <ol style="list-style-type: none"> Modify the RepositorySpecification entry to a point to a valid repository path. Example below. <pre> <Hive> <Repositories> <!--This is a list of the hive repositories on this server and their location--> <RepositorySpecification name="default" localPath="C:\Program Files\VMware\VCM\Tools\Repository\"> </Repositories> </Hive> </pre>

The Generate button is grayed out

Possible Cause	Remediation
The required fields are not yet populated with valid data.	Add valid data to the required fields, located on the Manage Packages Properties tab. The required fields are Name, Version, Architecture, Description.

Error when adding a Provides: the dependency package name is invalid

Possible Cause	Remediation
A Provides was defined with upper case letters. A current limitation allows you to add Provides containing upper case letters; however, Depends names allows only lower case letters. For example, the Provides in Package A is InternetExplorer, but when adding the Depends in Package B, you are only allowed to enter internetexplorer. This causes the Depends name to be invalid.	<p>If you can use Package Studio to repair the Provides in Package A (change it to internetexplorer) without reversioning the package, then do so and republish.</p> <p>If you cannot repair the Provides in Package A without reversioning, you can manually edit the dependencies in Package B.</p> <ol style="list-style-type: none"> 1. Locate the Package B .crate file. 2. Rename the file to .zip. 3. Unzip the files. 4. Locate and open the control.xml file with a text editing application. 5. Modify Depends entry referencing the Provides to contain the upper case Provides package name. For example, <code><Depends Name="internetexplorer" CrateOperator="LaterOrEqual" CrateVersion="7.0" /></code> is modified to read <code><Depends Name="InternetExplorer" CrateOperator="LaterOrEqual" CrateVersion="7.0" /></code>. 6. Save the control.xml. 7. Save and close the .zip file. 8. Rename the .zip to .crate.

When a package is created, signed, saved as project, and then generated, the package is no longer signed

Possible Cause

Remediation

By design, only generated packages are signed.

Generate the package before signing it.

The installed size of the application is set to nnn, but systems with nnn run out of space during installation

Possible Cause

Remediation

The assigned value nnn is the installed size, not the size required to unpack, download, and install the package contents.

Set the installed size to the amount of space required to install the package, even if the actual installed size is smaller.

The package installs the application but does not uninstall

Possible Cause

Remediation

The Removal options have not been configured in the package.

On the **Manage Packages** | **Files** tab, select Removal in the Installation/Removal drop-down list, and then configure any necessary Pre-Command, Commands, Arguments, and Post-Command options.

Files are added to the Project Data Directory, but they are not displayed in the list

Possible Cause

Remediation

Display needs to be refreshed.

Click the **Refresh Files List** button.

I am uncertain if the package requires a reboot to install or I only want the installation to reboot sometimes

Conditional reboots are not supported. You must either require a reboot or not. Whether or not an installation requires a reboot depends on running applications, state of the machines, the operating system, and other criteria. If you are uncertain if a reboot is required, but you do require a functioning system at the end of the installation or removal process, you should require a reboot.

Gathering Software Provisioning Logs on the Collector

Turn on the `AreResultsSaved` to view the following files from collections. See ["ARS Files" on page 21](#) for instructions for turning on the logs.

- **Provider.log:** Provides information on what the provider did, how it interacted with Package Manager (`wasp.exe`), and any problems encountered when formatting the results. The AgentBridge also writes to this log, recording any problems formulating provider instructions or transforming CDIF into element-normal xml.
- **CDIF files:** Provides the raw results written by the various providers. These files are useful in identifying special characters that are preventing the transformation element-normal xml.
- **RequestInterop.xml:** Provides a request document generated by the AgentBridge, which is used to invoke the providers. An error in the provider log will usually tell you which field you should evaluate.
- **Python output from Stdout:** Provides python output for both collections and actions. Stdout includes all provider/AgentBridge collection results and `wasp` output in an xml format.
- **Python output from Stderr:** Provides python output for both collections and actions. Stderr includes any result determined to be an error condition.

Gathering Software Provisioning Logs on the Agent

Modify the logging configuration file on the Agent.

1. Locate `Logging.conf`. The default location is `<Agent path>\Installers\Providers`.
2. Make a copy of `Logging.conf` so you can later restore the file.
3. Open the file with a text editor, such as Notepad.
4. Locate the section labeled `[handler_fileHandler]`.
5. Change the `args` setting in this area to the directory to which you want to write the logs. For example, `args=('c:\tmp\provider.log', 'a')`.
6. Run the problematic action from the Collector.
7. Restore the `Logging.conf` file to the previous state.
8. Review the log. It contains all the output from the pre-collection, the action, and the post-collection.

Turn on the Info logging for the Agent, allowing you to see how python is invoked in the Debut Event Log.

1. In VCM, select **Administration | Settings | Settings | Windows | Agent - General**. The **Agent General Settings** data grid appears.
2. Select **Type of information that should be logged**, and then click **Edit Settings**. The **Edit Settings** page of the **General Settings** wizard appears.
3. Select the following options: **Exception**, **Error**, and **Info**.

4. Click **Next**, and then click **Finish**.
5. Modify as needed and run the following on the command line of the Agent machine:

```
C:\WINDOWS\CMAgent\Installer\Python\python.exe -E
C:\WINDOWS\CMAgent\Installer\Providers\CommonPy\AgentBridge.py --
root=C:\WINDOWS\CMAgent\Installer\Providers --action-
template=C:\WINDOWS\CMAgent\Installer\Providers\Providers\Provisioning\Wasp\SourcesProvider\
RemoveRepositoryAction.template --action=template --
provider=Providers\Provisioning\Wasp\SourcesProvider\WaspSourcesProvider.py --
parameter=Uri="http://<RepositoryMachine>/SoftwareRepository" --
parameter=Platform="Any" --parameter=Section="Release" --parameter=request_
timeout_secs="28740"
```

6. You can add the `--temp-dir=c:\myProviderOutput` parameter to save all intermediate files and provider log to a specified directory. The directory must be created before you can use it. Specifying this directory will save all the intermediate files for actions, some of which are not available on the collector.
7. If you also include the `--parameter=LoginFile=`, you must modify the AgentBridge script to capture the file. For example, `--parameter=LoginFile="C:\WINDOWS\TEMP\LoginFile.enc"`. This file usually only exists for the duration of the job and this parameter is not optional.

Modify the argument handling for the 'parameter' argument in
 <path>\CMAgent\CommonPy\AgentBridge.py

```
elif o in ('--parameter='):
    param = str(a).strip("\"'")
    paramList = param.split('=', 1)
    key = paramList[0].strip("\"'")
    value = ''
    if len(paramList) > 1:
        value = paramList[1].strip("\"'")
    self.parameters[key] = value
    if key == 'LoginFile':
        from shutil import copy
        copy(value, 'c:\\myTemp\\')
```




Windows Agent Installation

Pre-Install Environment

Network

Pre-VCM Install Analysis

Required components and service for an Agent.

- IPC\$ must be available: used by RPC.
- Server Service must be available: Used by the collector to resolve the targets share to a local path for the purposes of registering the bootstrap service (Registration Service) with the Remote Service Control Manager (SCM).
- Ability to attach to the share with credentials provided:
- DNS must correctly resolve the machine\share to the appropriate machine.
- Authority to attach to the SCM: The authority of the collector service is used to attach to the SCM.

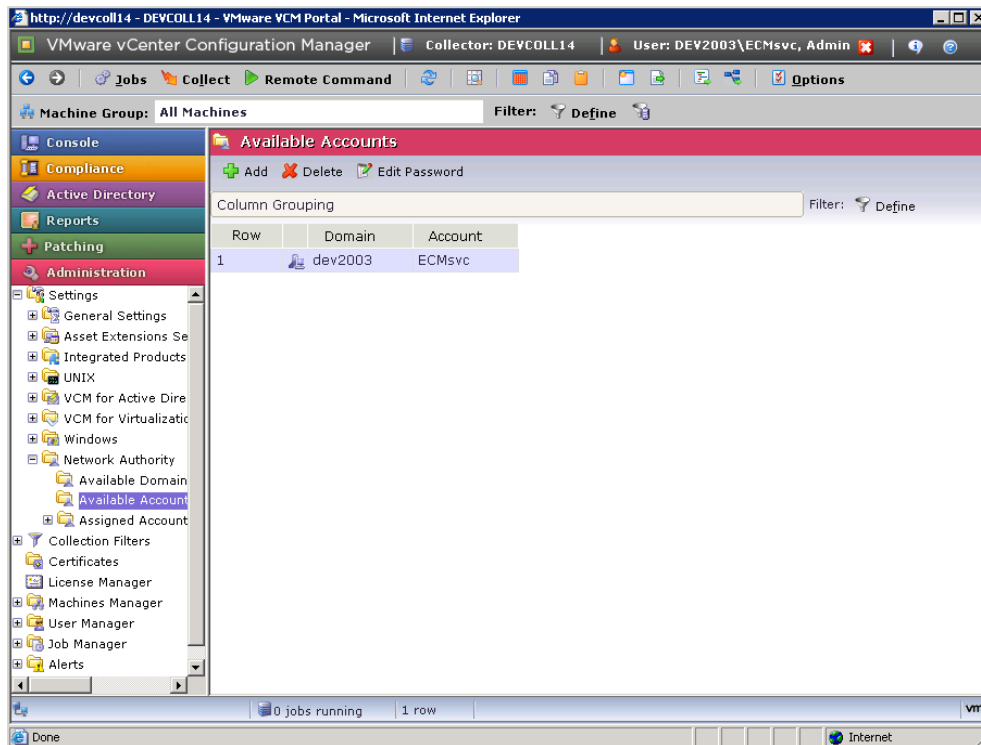
DCOM must be set to receive DCOM creation requests for various components. The following setting is required by DCOM.

- DCOM must be enabled on the target computer (On Windows 2003 server this option is turned off by default).
- On NT boxes, default security authorities (Access, Launch, and Configuration) must be explicit.
- Default authentication level is "Connect" and the default Impersonation level is "Identify".

Collector

Credentials/Authority Configuration

On the Collector the user defines domain admin account(s) from the Available Accounts data grid. These accounts are subsequently associated "By Domain" or "By Machine Group" with each targeted machine. The Collector allows multiple accounts to be specified in priority order. During installation the Collector cycles through these accounts as needed to establish communication with each target machine.



Install

The following describes in detail the install process from start to finish.

Upon issuing an installation request, looking at the Jobs Running dialog box, the following jobs are listed in the order that they are executed.

Job Summary: Install141624							
Refresh							
Step	Step Name	Waiting	Queued	Running	Succeeded	Failed	Total
1	Detect Previous Install	0	0	0	0	0	0
2	Validate Installation Environment	0	0	0	0	0	0
3	Interrogate Target Environment	0	0	0	0	0	0
4	Resolve Uninstall Dependencies	0	0	0	0	0	0
5	Uninstall Module	0	0	0	0	0	0
6	Uninstall Module Installer	0	0	0	0	0	0
7	Install Simple Installer	0	0	0	0	0	0
8	Install Module Installer	0	0	0	0	0	0
9	Resolve all versions of all modules based on highest version number	0	0	0	0	0	0
10	Install Module	0	0	0	0	0	0
11	Fully release the synchronization lock on the target machine	0	0	0	0	0	0
12	Submit request to Agent	0	0	0	0	0	0
13	Check if request is complete	0	0	0	0	0	0
14	Transfer request results	0	0	0	0	0	0
15	Acknowledge successful data transfer	0	0	0	0	0	0
16	Prepare request results for insert	0	0	0	0	0	0
17	Insert data into the database	0	0	0	0	0	0
18	Transform inserted data	0	0	0	0	0	0
19	Cleanup machine data	0	0	0	0	0	0
20	Partially release the synchronization lock on the target machine	0	0	0	0	0	0
21	Cleanup request data	0	0	0	0	0	0

1. **Detect Previous Install (Detects previous Agents and Agent components):** Detect previous install determines if a previous Agent version is installed on the target machine. When installing 4.11.x or later, if a previous Agent version is detected, it must be removed. The Detect Previous Install process determines if a previous Agent is present by attempting to connect to the agent installation DCOM components (Basic and Agent Installers). If a connection is made to either of these components, Detect Previous Install sets a state member that the following three steps use to determine if they should execute. This step and the three that follow it remove the previous Agents and are included in all types of install and uninstall requests.
2. **Validate Installation Environment**
 - a. The validate installation environment job is the first real job that gets executed. This job ensures that the target is available for installation. If the validate job is able to contact the Module Installer, it calls methods on that component to get all the install manifests on the target. It also checks the registry for various conditions that should prevent the install. This job fails if the following conditions are satisfied:
 - **The agent is locked:** The following shows the registry entry that is evaluated by this job to determine if the agent is locked. HKEY_LOCAL_MACHINE\SOFTWARE\Configuresoft\ECM\4.0\Agent\IsLocked. If this key exists and has a value other than 0 then the agent is locked.

NOTE The install infrastructure cannot remove or modify this value.

 - **The Agent is a Collector:** The following shows the registry entry that is evaluated by this job to determine if the agent is on a collector box HKEY_LOCAL_MACHINE\SOFTWARE\Configuresoft\ECM\4.0\Installer\CDInstall. If this key exists and has a value of 2 then this identifies the box as a collector box. **NOTE:** The install infrastructure cannot remove or modify this value.
 - b. If the validate job is not able to communicate with the install infrastructure, it flags the install as valid and the request continues.
 - c. If an error, an error would be any result other than Install required or Success, is generated by an attempt to communicate with the install infrastructure, validate fails, as does the install, usually with an error that mandates that an uninstall must occur prior to installing.
 - d. If the validate job is able to communicate with the Module installer, it gathers all the module information available from the various install manifests it gathered, and stores this information as a state variable so that other install jobs can use the information to see what was installed. It does this in the method RecordModuleInstallerVersion, and stores the module names and versions in the state parameter "install_module". The module installer job uses this parameter to determine what parts of the install infrastructure need to be installed.
3. **Interrogate Target Environment:** After the installation infrastructure has been deployed, the various product modules need to be deployed. In order to find out what product modules need to be pushed out, the following process occurs.
 - a. A connection is made to the module installer.
 - b. The runtime agent lock is updated.
 - c. The module installer returns all modules currently installed.
 - d. The product modules are then recorded in the database.
 - e. A failure here will cause install to fail.

4. **Resolve Uninstall Dependencies:** Based on what was found on the target and what is to be deployed, reconciliation occurs. The result of this is a list of what product modules may need to be uninstalled and a list of product modules to install. Any product modules that require deploying are recorded in the state matrix. A failure here will cause installation to fail.
5. **Uninstall Module:** All product modules that need to be uninstalled are managed here. The following is a brief description of this process.
 - a. A list of product modules is obtained from the database.
 - b. A connection is made to the module installer.
 - c. The runtime agent lock is updated.
 - d. Each module is uninstalled in turn by the module installer.
 - e. If an error occurs during this process, installation fails.
6. **Uninstall Module Installer**
 - a. This job is part of the complete removal of an agent. If you are uninstalling the agent, this job tries every means possible to completely remove the agent.
 - Via asynchronous command using the Registration Service.
 - Via command to simple installer
 - Via remote share
 - b. If the current request is for any other kind of install action, this is a no-op.
7. **Install Simple Installer**
 - a. Install Simple Installer lays down the installation infrastructure. It uses a service to deploy the installation infrastructure modules. Install infrastructure modules included at this time are ECMSimple.exe, ECMSimpleInstaller.exe, and ECMComSocketListener.exe. The following describes the actions that are preformed during this step.
 - b. An attempt is made to contact the Simple Installer on the target machine. If this action is successful then this step is complete, else the following occurs:
 - c. Attempt to attach to the share on the target that was specified in the UI and using the specified authority. If this fails, then the installation fails.
 - d. If the attach to share succeeded, the next step is to see if the Agent has a runtime lock. This lock (not the same as the IsLocked registry entry), is used by the installation process to prevent two or more Collectors from installing on the same target to the same location at the same time. The lock manifests itself as a file on the target named (ECMMachineActionLock.dat). This lock file is located in the root directory of the install and contains the type of action being performed (installing, uninstalling, collecting), a time stamp, and the request id of the action doing the work. If the lock file exists on the target, contains a valid action with a request id, and contains a time stamp that has not expired, the target is then considered locked and the install fails. In this case the lock is partially unlocked (the time stamp is zeroed and the request id is purged). This allows a subsequent install to continue provided it was not relocked by another Collector. In all other cases the lock file is zeroed and the install is allowed to continue.
 - e. A pre-installation check is then preformed to ensure that the subsequent install will succeed. If various components (ComSocketServiceListener, debug event dll, or the subsystem singleton dll) are located, they are removed.

- f. The directory structure is created and the following components are laid down on the target.
 - ECMSCollInstallAgtRegistrationService.exe – This binary is the bootstrap service that will be invoked by the target's Remote Service Control Manager (SCM).
 - Psapi.dll – This file is a dependency of the Registration Service.
 - ECMTARGETShareInfo.dat – This file is generated and contains the share being used for this install as well as the relative path to the agent.
 - g. Attempt to connect to the SCM using the same authority as the Collector service itself. If this fails then the installation fails.
 - h. The ECMSCollInstallAgtRegistrationService.exe is then registered with the target the SCM.
 - i. The following loop executes until all the install infrastructure modules are deployed:
 - Copy an install infrastructure module to the target.
 - Start the registration service. This service executes the module and waits until it is fully expanded.
 - If the module reports an error, terminate the install. (A log file is left on the target specifying the nature of the problem.)
 - Stop the service.
 - j. The registration service is unregistered with VCM.
 - k. An attempt is made to contact the Simple Install via DCOM. If this succeeds then this step is complete.
 - l. If at any time the process fails, the simple installer job enters into its rollback state. All modules are uninstalled, all files are removed, and the registry is purged. In this state debug event file are not deleted on the target for they may contain useful information on the nature of the error.
8. Install Module Installer
- a. The Module installer deals with installing various product modules. It is called upon to terminate agent process when an inspection job is canceled and is able to update the install infrastructure. The module installer is responsible for completely removing the remainder of VCM during uninstall.
 - b. When this step is executed its first job is to determine if the module installer component has been deployed. If it exists a check is preformed to see if it needs to be upgraded.
 - c. If the module installer needs to be installed:
 - A connection is made to the simple installer
 - The runtime install lock is updated.
 - The module installer is pushed out and executed by the simple installer.
 - d. The next step is to determine if any install infrastructure modules need to be upgraded. A list is made of those outdated components and they are upgraded.

- e. If a component (other than the module installer) of the installation infrastructure needs to be updated the following actions occur.
 - A connection is made to the module installer.
 - The registration service is copied if it is not present on the target.
 - All infrastructure modules that need to be updated are copied to the target.
 - A call is made to the module installer to install the various install infrastructure modules. This call is then received and delegated to the registration service to do the work.
- f. Finally, if any of the install infrastructure modules need to be removed then they are removed.
- g. If at any time a failure is realized, this step rolls back all of the work it had done and the install fails.
9. Resolve all versions of modules based on highest version number
 - a. This job looks at the modules being installed and those already on the agent and determines exactly what needs to be installed.
 - b. Resolve Highest Version Modules Resolution Algorithm: The Resolve Highest Version Modules Resolution Algorithm is presented with a Requested Install Module List (RIML), the Already Installed Module List (AIML), the Module Dependency Graph, and the Module Updates Mapping. This algorithm determines what needs to be installed and uninstalled on the agent to achieve the highest versions of modules based on the modules provided in RIML and AIML.
10. Install Module: All product modules needing to be installed are installed at this point in the process. The following is a brief description of this process.
 - a. A list of product modules is obtained from the database.
 - b. A connection is made to the module installer.
 - c. The runtime agent lock is updated.
 - d. Each module is installed by the module installer.
 - e. If an error occurs during this process, install fails
11. Fully release the synchronization lock on the Agent.
 - a. If this job is executed, (note a failure from any step listed above causes all previous steps to skip) the runtime agent lock is completely cleared. This allows an Agent to inspect.
 - b. If this job is not executed (due to a previous failure), the installation is considered invalid. An attempt to collect from an agent will result in failure since the install failed.
 - c. The following describes this process.
 - A connection is made to the module installer.
 - Unlock is called.
12. Submit Request to agent: This step and the steps that follow deal with Agent inspection. After an install, a machine environment inspection is performed. Detailed information lies outside the scope of this document.
13. Check if Request is Complete
 - a. The Collector checks to see if the machine environment collection is complete.
 - b. Detailed information lies outside the scope of this document.

14. Transfer Request Results
 - a. Upon the completion of the machine environment inspection, the results are sent back to the collector.
 - b. Detailed information lies outside the scope of this document.
15. Acknowledge successful data transfer
 - a. Collector records the fact that it got all the data from the machine environment inspection.
 - b. Detailed information lies outside the scope of this document.
16. Prepare request results for insert.
 - a. A bulk insert is prepared.
 - b. Detailed information lies outside the scope of this document.
17. Insert data into database
 - a. The data from the machine environment is inserted.
 - b. Detailed information lies outside the scope of this document.
18. Transform inserted data.
 - a. The data is transformed from temp tables and meta data is updated
 - b. Detailed information lies outside the scope of this document.
19. Cleanup request data
 - a. Collector is cleaning up.
 - b. Detailed information lies outside the scope of this document.
20. Partially release the synchronization lock on the target machine.
 - a. This step, known as a finalize job, is always run even if previous steps fail.
 - b. The goal of this step is to clear out the Agent runtime lock to the extent that if another install is attempted it will succeed. This step does not clear the action type in the lock. This partial release has the effect of rendering the Agent incapable of doing useful work, since this condition is a result of an invalid install. Only if step 11 is executed is the agent available for collections.

Uninstall

Uninstall is the process of removing VCM Agent presence from a target box. This removes VCM products and the install infrastructure. The result of uninstall is a clean box with no VCM remnants.

Upon issuing an installation request, looking the Jobs Running dialog box, the following jobs are listed in the order that they are executed.

Job Summary: UnInstall182608							
Refresh							
Step	Step Name	Waiting	Queued	Running	Succeeded	Failed	Total
1	Detect Previous Install	0	0	0	0	0	0
2	Validate Installation Environment	0	0	0	0	0	0
3	Interrogate Target Environment	0	0	0	0	0	0
4	Resolve Uninstall Dependencies	0	0	0	0	0	0
5	Uninstall Module	0	0	0	0	0	0
6	Uninstall Module Installer	0	0	0	0	0	0
7	Fully release the synchronization lock on the target machine	0	0	0	0	0	0
8	Partially release the synchronization lock on the target machine	0	0	0	0	0	0
9	Cleanup request data	0	0	0	0	0	0

1. Detect Previous Install is described in detail under the Install section, step 1.
2. Validate installation environment is described in detail under the Install section, step 2.
 - Note all steps after validate will be executed. Uninstall mandates that each job should be run regardless of failure.
3. Interrogate target environment is described in detail under the install section, step 3.
4. Resolve uninstall dependencies.
 - The resolve uninstall dependencies step uninstalls all the product modules on the target box. Product modules are reconciled and flagged for removal and recorded in the data base.
5. Uninstall Module is described in detail under the install section, step 5
6. Uninstall module installer:
 - a. The uninstall module step is responsible for the bulk of the work during uninstall. After this step is executed any and all remaining product and install infrastructure modules are removed. The following describes this step in more detail:
 - A connection is made to the module installer.
 - A complete uninstall request is generated and sent to the module installer.
 - The module installer copies the registration service to the users temp directory.
 - It then calls upon the registration service to perform the complete uninstall and the module installer terminates itself.
 - The registration service (running as an exe now). Iterates over the agent's directories making note of all installed modules.
 - It then in turn uninstalls each module it finds.
 - The registration service ensures that all VCM files are removed from the file system and that the registry is purged.
 - It then marks itself for deletion upon the next reboot of the computer.

- b. The next step is an attempt to attach to the target machine's share. This attempt is made even if the previous step succeeds. It may or may not work since we do not guarantee the ability to attach to a share during uninstall, but if it can attach, it removes all the VCM files that it finds base on a filter stored in the database.
 - c. Uninstall then tries to connect to the remote registry. If it can, it removes all the VMware registry entries it finds.
7. Fully release synchronization lock: If uninstall worked correctly, a connection cannot be made with the module installer there is no further action needed to fully release synchronization lock. However, if uninstall fails and the module installer can be reached the runtime lock is released.
8. Partially release the synchronization lock on the target machine: If uninstall worked this step is a no-op else the runtime action lock time stamp and the request id are cleared.
9. Cleanup request data: The request is removed and the state of this request is complete.

Upgrade

Upgrade is the process by which various modules, either install infrastructure or product, may be upgraded. This request is identical to an Agent install request with the single caveat that an Agent must be present in order to upgrade. Beyond that, see the Install section for details.

Change Protocol Request

A change protocol request is applied when you wish to change the protocol the Collector uses when communicating with a machine. Currently there are two protocols used for communicating with agents: HTTP and DCOM. DCOM is historically our oldest supported protocol and dates back to the 3.0x version of the VCM product. HTTP protocol was introduced with 4.5. A change protocol request works as follows:

- When changing the protocol from DCOM to HTTP, the EcmComSocketListenerService module is shipped over and executed.
- When changing from HTTP to DCOM, the EcmComSocketListenerService is removed. The following is an example of changing protocol.

Job Summary: ChangeProtocolToDCOM					
Refresh					
Step	Step Name	Queued	Running	Succeeded	Failed Total
1	Detect Previous Install	0	0	1	0 1
2	Uninstall Agent	0	0	1	0 1
3	Uninstall Package Installer	0	0	1	0 1
4	Uninstall Basic Installer	0	0	1	0 1
5	Validate Installation Environment	0	0	1	0 1
6	Install Simple Installer	0	0	1	0 1
7	Store installation data in the database	0	0	1	0 1
8	Install Module Installer	0	1	0	0 1
9	Fully release the synchronization lock on the target machine	0	0	0	0 0
10	Submit request to Agent	0	0	0	0 0
11	Check if request is complete	0	0	0	0 0
12	Transfer request results	0	0	0	0 0
13	Acknowledge successful data transfer	0	0	0	0 0
14	Prepare request results for insert	0	0	0	0 0
15	Insert data into the database	0	0	0	0 0
16	Transform inserted data	0	0	0	0 0
17	Cleanup machine data	0	0	0	0 0
18	Partially release the synchronization lock on the target machine	0	0	0	0 0
19	Cleanup request data	0	0	0	0 0

1. Detect Previous Install is described in detail under the Install section, step 1.
2. Uninstall Agent is described in detail under the Install section, step 2.
3. Uninstall package installer is described in detail under the Install section, step 3.
4. Uninstall basic installer is described in detail under the Install section, step 4.
5. Validate installation environment is described in detail under the Install section, step 5.
6. Install Simple Installer: During a change protocol request, this job receives special parameters that describe how the Com Socket Service Listener module should be handled. If the change protocol request was from DCOM to HTTP, the simple installer component receives a request to install the ComSocketServiceListener. If the change protocol request was from HTTP to DCOM the simple installer receives a request to remove the ComSocketServiceListener. It is important to note that the simple installer itself cannot modify the contents of the install infrastructure. The simple installer job is only used to initially deploy the infrastructure using the Registration service as a bootstrap loader. Since the simple installer job cannot take action on the information it receives it delegates the work to the Install Module Installer step.
7. Store installation data in the database described in detail under the Install section, step 7.
8. Install Module Installer step is where the work really happens with a change protocol request. Since the module installer has all the information of what install infrastructure modules were installed on the box, and has the information concerning if the ComSocketServiceListener should be installed or removed, it is able to reconcile the information and find out what needs to be done. If the ComSocketServiceListener should be installed, it bundles up a request and sends the work of installing this module to the Registration Service. Uninstalling the ComSocketServiceListener happens in a similar fashion.
9. Fully release the synchronization lock on the target machine is described in detail under the Install section, step 13.
10. Steps 10 – 19 are the same as listed in the install section

Manual Agent Install

Manual agent install is the process by which the Agent is deployed to the target without the use of a Collector. The main goal of manual install is to produce an Agent environment that is identical to an Agent environment that one would expect if a Collector deployed the agent. Manual install does not create entries in the “add and remove programs section” nor is it designed to be uninstalled manually. Manual install takes on two forms. The first is the standard “agent only” option on the VCM Product installation CD. The second is the use of the actual Wise binary.

Install

Manual agent install is essentially a group of modules deployed with the use of Wise installation system. The VCM Product installation CD is an Install Shield program. If one selects “Agent Only” install, then Install Shield delegates the work of installing the Agent to the executable CMAgentInstall.exe. This executable, built by Wise Installation Systems 9 is the manual install program. Regardless of how it is executed, it is the program that lays down the Agent. This executable may be used interactively or silently. The following describes each usage.

1. Interactive: Interactive mode displays various dialog boxes for input regarding the following pieces of information that the system needs to lay down the agent.
 - a. Installation directory – In interactive mode the user must specify the directory that the agent is to be installed to.
 - b. Lock agent – If the agent is to be locked, then this box is to be checked.
 - c. HTTP – If this is to be an HTTP agent installation, this box is checked the user must provide a port that the agent will use to receive request.
 - d. In interactive mode, if the user specifies an HTTP port already in use by the system, the program prompts you with another selection. This process continues until an open port is found.
 - e. Once the following conditions are met the installation program lays down all the install, product, and upgrade modules associated with the Agent.
2. Silent: The CMAgentInstall.exe program may be executed silently. This is available to facilitate alternate methods of Agent deployment. For example, a common way to deploy an Agent is to create a script that runs on the host box and executes the manual Agent install program silently. To install the manual install program via a command line, the following options are available.
 - Silent mode switch is /S.
 - To install a DCOM agent silently, the command line is: CMAgentInstall.exe /S INSTALLPATH=C:\MyVCM. Where INSTALLPATH is the destination of the agent files.
 - To install an HTTP agent silently, the command line is: CMAgentInstall.exe /S INSTALLPATH=C:\MyVCM PORT=26542. Where PORT is the port that the agent is to use. In this mode if the port is in use, manual install fails. The path for CMAgentInstall.exe may be a UNC path.
3. Manual install essentially executes the same modules that the collector would to install an agent. The following modules are executed by CMAgentInstall.exe.
 - a. ECMNotUpdateable.exe
 - b. ECMSimpleInstaller.exe
 - c. ECModuleInstaller.exe
 - d. ECMCommon.exe
 - e. Optionally, CsiWin32SocketListener.exe

Uninstall

It must be pointed out that although it is possible to uninstall the agent manually, it is prone to error and not recommended! The intent of a manual install is to install, not to uninstall.

Upgrade

A manual upgrade is not supported.

Protocol Specific

Protocol plays a fundamental role when installing, uninstalling, and upgrading. Protocol availability is dictated on the agent by the existence of the install infrastructure module named CsiWin32SocketListener.exe. This module contains the http service that listens to Collector requests on a given port. The service is thin and delegates request to components that are part of the standard install. The Collector is responsible for determining the protocol to talk to the agent with. An HTTP agent can be

accessed via a port or via DCOM, where a DCOM agent can only respond to DCOM requests.

DCOM

Install happen via DCOM. Currently there are not other methods. Uninstall and upgrades are not bound by this limitation.. DCOM is also the lowest common protocol used for collections and installations. If the Collector lists an Agent as listening with HTTP, and the HTTP connection cannot be established, DCOM is attempted.

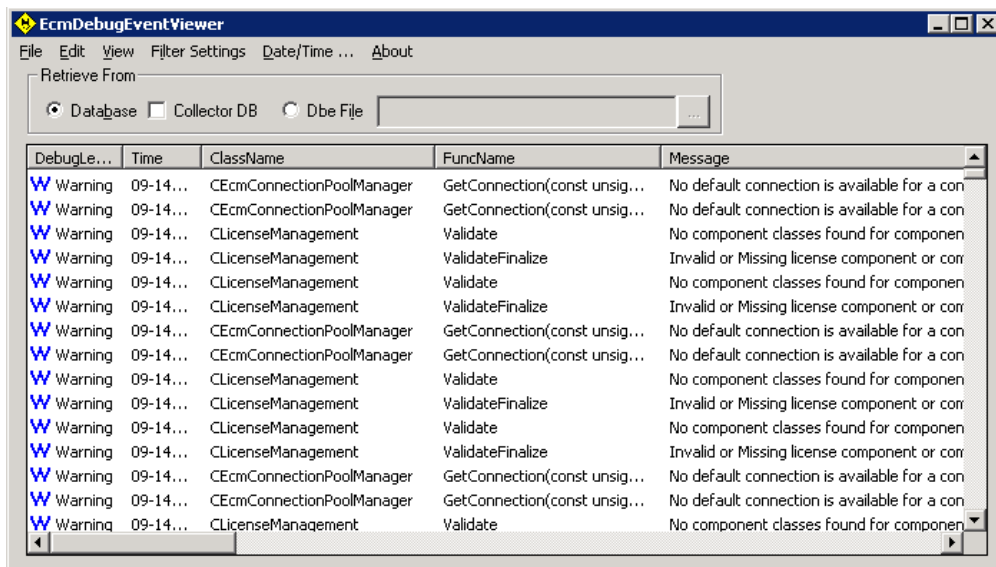
HTTP

HTTP was added to VCM as an alternative to DCOM in all cases except install. HTTP is the addition of the CsiWin32SocketListener.exe module on the agent.

Debugging Tip(s)

Module Management processing provides a wealth of debugging information as each module resolution algorithm is executed by the collector on a per agent basis. Use this detailed information when debugging problems related to module resolution or to gain insight on the impact that installs or uninstalls have on a particular Agent. Each resolution algorithm will dump the current and requested Agent modules, module dependency graph and the module updates mapping prior to algorithm execution as well as each algorithms step-by-step output as it is executes.

Example debug (INFO) output captured during an agent install:



UNIX Agent Troubleshooting

Agent Directory Structure

Assuming that the defaults were taken during install, executing the command `ls -laR` will return the following listing. Comments have been added to help in reading this file information.

The `/opt` directory is the default install location.

inetd installation:

```
/opt/:
dr-xr-x--x  9 root      cfgsoft      320 May  6 16:47 CMAgent
```

The CMAgent sub-directory is the root of the Agent installation. The CSIRegistry file is an XML file representing the configuration data for the agent. Conceptually, it acts like the Windows Registry. Note that all of the files are owned by root and the cfgsoft group.

```
/opt/CMAgent:
drwxr-x---  3 root      cfgsoft      112 May  6 16:47 Agent
drwxr-x---  3 root      cfgsoft      112 May  6 16:47 CFC
-rw-rw----  1 root      cfgsoft 50455 May  8 12:20 CSIRegistry
drwxrwx---  3 root      cfgsoft      136 May  6 16:47 ECMu
dr-xr-x--x  3 root      cfgsoft      120 May  6 16:47 ThirdParty
drwxrwx--- 11 csi_acct  cfgsoft      368 May  8 12:18 data
drwxr-x---  6 root      cfgsoft      528 May  6 16:47 install
lrwxrwxrwx  1 root      cfgsoft        20 May  6 16:47 log ->
/var/log/CMAgent/log
drwxrwx---  2 root      cfgsoft        80 May  6 16:47 uninstall
```

The Agent sub-directory contains code libraries that are specific to the agent. It contains a sub-directory for the code version (3.0) which in turn contains a lib directory with all libraries and a bin directory with all executables.

```
/opt/CMAgent/Agent:
```

```

drwxr-x---  4 root      cfgsoft    96 May  6 16:47 3.0
-rw-r----- 1 root      cfgsoft  3203 May  6 16:47 manifest_
Agent.3.0.Linux

/opt/CMAgent/Agent/3.0:
drwxr-x---  2 root      cfgsoft    48 May  6 16:47 bin
drwxr-x---  2 root      cfgsoft  2432 May  6 16:47 lib

/opt/CMAgent/Agent/3.0/bin:
<--- currently empty ---->

/opt/CMAgent/Agent/3.0/lib:
-r-xr-x---  1 root      cfgsoft  246584 May  6 16:47
libAgentFileManagerSubsystem.so
-r-xr-x---  1 root      cfgsoft   76696 May  6 16:47 libAgentResponse.so
-r-xr-x---  1 root      cfgsoft  106460 May  6 16:47 libChangeCommon.so
-r-xr-x---  1 root      cfgsoft  102024 May  6 16:47
libChangeFactorySubsystem.so
-r-xr-x---  1 root      cfgsoft   28192 May  6 16:47 libCommonQueues.so
<---snip--->
-r-xr-x---  1 root      cfgsoft  287320 May  6 16:47
libStateMachineDefinitions.so
-r-xr-x---  1 root      cfgsoft  295940 May  6 16:47
libStateMachineEngine.so
-r-xr-x---  1 root      cfgsoft  250800 May  6 16:47
libStateMachineFactorySubsystem.so
-r-xr-x---  1 root      cfgsoft  964456 May  6 16:47 libStatusManager.so

```

The CFC sub-directory contains code libraries that are common components. It contains a sub-directory for the code version (3.0) which in turn contains a lib directory with all libraries and a bin directory with all executables.

```

/opt/CMAgent/CFC:
drwxr-x---  4 root      cfgsoft    96 May  6 16:47 3.0
-rw-r----- 1 root      cfgsoft  2677 May  6 16:47 manifest_CFC.3.0.Linux

/opt/CMAgent/CFC/3.0:
drwxr-x---  2 root      cfgsoft   312 May  6 16:47 bin
drwxr-x---  2 root      cfgsoft  1888 May  6 16:47 lib

```



```
/opt/CMAgent/CFC/3.0/bin:
```

```
-r-xr-x--- 1 root      cfgsoft 107517 May  6 16:47 CSI_
ManageCertificateStore
-r-xr-x--- 1 root      cfgsoft   87490 May  6 16:47 CsiAgentListener
-r-xr-x--- 1 root      cfgsoft  290823 May  6 16:47
CsiListenerWorkerDaemon
-r-xr-x--- 1 root      cfgsoft   15324 May  6 16:47 RegisterSubSystem
-r-xr-x--- 1 root      cfgsoft   27080 May  6 16:47 RegistryAdd
-r-xr-x--- 1 root      cfgsoft   17404 May  6 16:47 RegistryRead
-r-xr-x--- 1 root      cfgsoft   15176 May  6 16:47 UnregisterSubSystem
```

```
/opt/CMAgent/CFC/3.0/lib:
```

```
-r-xr-x--- 1 root      cfgsoft 1873024 May  6 16:47
libCfcCommonAggregator.so
-r-xr-x--- 1 root      cfgsoft   400160 May  6 16:47
libCfcDataSerializable.so
-r-xr-x--- 1 root      cfgsoft   179776 May  6 16:47
libCfcDebugEventSubSystemSingleton.so
-r-xr-x--- 1 root      cfgsoft   633924 May  6 16:47 libCfcEncoding.so
<---snip--->
-r-xr-x--- 1 root      cfgsoft    71996 May  6 16:47
libSynchronization.so
-r-xr-x--- 1 root      cfgsoft  249596 May  6 16:47 libThreadPool.so
-r-xr-x--- 1 root      cfgsoft  316328 May  6 16:47 libUNIXIPCCore.so
-r-xr-x--- 1 root      cfgsoft  431640 May  6 16:47 libXMLParser.so
```

The location of the data directory is configurable at install time. It contains all of the inspection results, master files, etc., and is used for any temporary files that are created during the inspection process.

```
/opt/CMAgent/data:
```

```
drwxrwx--- 4 csi_acct      cfgsoft   96 May  8 14:11 <Database_name>
drwxrwx--- 5 root          cfgsoft  152 May  8 12:28 db
drwxrwx--- 4 csi_acct      cfgsoft  152 May  7 11:53 tmp
```

The db directory contains sub-directories for the data model and status manager Birdstep databases. It also contains a sub-directory for the agent's certificate store

```
/opt/CMAgent/data/db:
```

```
drwxrwx---  3 root          cfgsoft  72 May  6 16:47 DtmDB
drwxrwx---  3 root          cfgsoft 104 May  6 16:47 PDS
drwxrwx---  3 root          cfgsoft  72 May  6 16:47 SM
```

```
/opt/CMAgent/data/db/DtmDB:
```

```
drwxrwx---  8 root          cfgsoft 320 May  8 12:19 RDM
```

The DtmDB/RDM directory is the Birdstep database containing the default data model. This data model forms the basis for replicated data models from the agent as described later.

```
/opt/CMAgent/data/db/DtmDB/RDM:
```

```
-rw-rwx---  1 root          cfgsoft 1865 May  6 16:47 DtmDB.dbd
-rw-rwx---  1 root          cfgsoft 2048 May  6 16:47 DtmKeys.dbd
-rw-rwx---  1 root          cfgsoft 9216 May  6 16:47 DtmRecord.dbd
```

The SM/RDM contains the Birdstep database which holds information about running requests, average length that a request has executed etc. Development has a tool which can be used to convert this file to an XML representation for debugging purposes. This tool is not delivered with the agent as it would expose proprietary information.

```
/opt/CMAgent/data/db/SM/RDM:
```

```
-rw-rwx---  1 root          cfgsoft  7680 May  8 12:19
MachineStatusLog.dbd
-rw-rwx---  1 root          cfgsoft   2048 May  8 12:19 MachineStatusLog_
K1.dbd
-rw-rwx---  1 root          cfgsoft 10752 May  8 12:20
RequestStatusLog.dbd
-rw-rwx---  1 root          cfgsoft   2048 May  8 12:19 RequestStatusLog_
K1.dbd
-rw-rw----  1 csi_acct      cfgsoft    498 May  8 12:28 SM.taf
-rw-rwx---  1 root          cfgsoft 268800 May  8 12:19
StateMachineStateHistoryLog.dbd
-rw-rwx---  1 root          cfgsoft  41984 May  8 12:19
StateMachineStateHistoryLog_K1.dbd
-rw-rwx---  1 root          cfgsoft  12800 May  8 12:19
StateMachineStatusLog.dbd
-rw-rwx---  1 root          cfgsoft   2048 May  8 12:19
StateMachineStatusLog_K1.dbd
```

```
-rw-rwx--- 1 root      cfgsoft  2361 May  6 16:47
StatusManagerDB.dbd
```

The PDS directory contains the Certificate Store.

```
/opt/CMAgent/data/db/PDS:
-rw-rw---- 1 root      cfgsoft 43215 May  6 16:47 CertStore
drwxrwx--- 2 root      cfgsoft   80 May  6 16:47 sib
```

These files **must** be readable by the cfgsoft group in order for the agent to validate a collector. If the agent package was copied from a collector, then the certificate for the collector will be preloaded into the Certificate Store. Other certificates may be added using the [CSI_ManageCertificateStore](#) executable.

```
export LD_LIBRARY_
PATH=/opt/CMAgent/CFC/3.0/lib:/opt/CMAgent/ThirdParty/1.0/lib
export CSI_REGISTRY_PATH=/opt/CMAgent
/opt/CMAgent/CFC/3.0/bin/CSI_ManagerCertificateStore -i -f <cert-file>
```

Note: Replace LD_LIBRARY_PATH with LIBPATH for AIX; SHLIB_PATH for HP-UX; DYLD_LIBRARY_PATH for Mac OS as necessary.

The ECMu sub-directory contains code libraries that are specific to the Unix Agent. It contains a sub-directory for the code version (1.0) which in turn contains directories with all of the binaries and scripts for ECMu.

```
/opt/CMAgent/ECMu:
drwxrwx--- 7 root      cfgsoft  176 May  7 11:53 1.0
-rw-r----- 1 root      cfgsoft 1333 May  6 16:47 manifest_
ECMu.1.0.Linux
-rw-r--r-- 1 root      cfgsoft   4 May  6 16:47 version

/opt/CMAgent/ECMu/1.0:
drwxrwx--- 2 csi_acct  cfgsoft   80 May  7 11:53 RDM
drwxr-x--- 2 root      cfgsoft  240 May  6 16:47 bin
drwxr-x--- 2 root      cfgsoft  624 May  6 16:47 lib
drwxr-x--- 2 root      cfgsoft  184 May  6 16:47 registration
drwxr-x--- 2 root      cfgsoft  376 May  6 16:47 scripts
```

There are three files in the bin directory that are used when running inspections and remote commands.

- **RunHigh:** Used to run privileged inspections. This is achieved because it is owned by root and has the suid permission set (note the r-s in the permissions)
- **RunLow:** Used to run unprivileged inspections. This is achieved because it is owned by the primary group (nobody) of the user that the agent runs as and has the sgid permission set. When this program runs it switches to the nobody group and cannot execute any commands that require root privilege.
- **RunRemote:** Used to execute privileged Remote Commands and operates in the same manner as RunHigh.

If these executables fail, they will log errors of type auth.err to /var/log/secure (Linux) or /var/adm/messages (Solaris) or to wherever these messages types are configured to be logged in /etc/syslog.conf. The log message merely states that they failed. It will not identify the reason for failure as a hacker could use the information to break the security of the programs.

Development has versions of these programs which will log an error code as well as the message. The error code can then be used to determine the reason for the failure by inspection of the source code.

```
/opt/CMAgent/ECMu/1.0/bin:
```

```
-r-xr-x--- 1 root      cfgsoft 114023 May  6 16:47 Agent
-r-sr-x--- 1 root      cfgsoft  11375 May  6 16:47 RunHigh
-r-xr-s--- 1 csi_acct  csi_acct 11440 May  6 16:47 RunLow
-r-sr-x--- 1 root      cfgsoft   9686 May  6 16:47 RunRemote
-r-xr-x--- 1 root      cfgsoft 124059 May  6 16:47 TestMetadata
-r-xr-x--- 1 root      cfgsoft  53341 May  6 16:47 cabextract
-r-xr-x--- 1 root      cfgsoft 111994 May  6 16:47 csipccli
```

```
/opt/CMAgent/ECMu/1.0/lib:
```

```
-r-xr-x--- 1 root      cfgsoft  13216 May  6 16:47 libAgentXPCommon.so
-r-xr-x--- 1 root      cfgsoft 315140 May  6 16:47
libAwkScriptDriverSubsystem.so
-r-xr-x--- 1 root      cfgsoft 249236 May  6 16:47
libAwkScriptDriverSyslogEventsSubsystem.so
-r-xr-x--- 1 root      cfgsoft 544880 May  6 16:47 libCsiPpxLibHelper.so
-r-xr-x--- 1 root      cfgsoft 360720 May  6 16:47
libEcmAgentInspectorCommon.so
-r-xr-x--- 1 root      cfgsoft  64352 May  6 16:47
libEcmAgentInspectorScript.so
-r-xr-x--- 1 root      cfgsoft 274360 May  6 16:47
libEcmFileUploadJob.so
-r-xr-x--- 1 root      cfgsoft 226240 May  6 16:47
libEcmRemoteCommandJob.so
```

```

-r-xr-x--- 1 root      cfgsoft 204968 May  6 16:47
libEcmScriptInspectionJob.so

-r-xr-x--- 1 root      cfgsoft  64392 May  6 16:47
libPatchFactorySubsystem.so

-r-xr-x--- 1 root      cfgsoft 138424 May  6 16:47
libScriptChangeStateMachineJob.so

-r-xr-x--- 1 root      cfgsoft 167384 May  6 16:47
libXpChangeDriverState.so

/opt/CMAgent/ECMu/1.0/registration:

-rw-r----- 1 root  cfgsoft 2117 May  6 16:47 CMAgent.rpm
-r-xr-x--- 1 root  cfgsoft 2299 May  6 16:47 RegisterAgent.sh
-r-xr-x--- 1 root  cfgsoft 1044 May  6 16:47 UnregisterAgent.sh
-rw-r----- 1 root  cfgsoft  620 May  6 16:47 cmagent.deb

```

The scripts sub-directory contains scripts that are used when running the agent. The file `csi-agent` is a copy of the details installed into the (x)inetd configuration. The `inetd-agent` is the script that (x)inetd invokes when an attempt is made to contact the agent.

The `stopagent.sh` script is provided as a means to manually kill all of the Agent processes in a clean manner.

```

/opt/CMAgent/ECMu/1.0/scripts:

-r-xr-x--- 1 root      cfgsoft      351 May  8 13:02 boot-init.sh
-r--r----- 1 root      cfgsoft     1279 May  8 13:02 boot-init.sh.lsb
-r--r----- 1 root      cfgsoft     1317 May  8 13:02 boot-init.sh.RH
-r--r----- 1 root      cfgsoft     3375 May  8 13:02 boot-
init.sh.SuSE
-r--r----- 1 root      cfgsoft       75 May  8 13:02 csi-agent
-r--r----- 1 root      cfgsoft      249 May  8 13:02 csi-agent-xinetd
-r-xr-x--- 1 root      cfgsoft      263 May  8 13:02 inetd-agent
-r-xr-x--- 1 root      cfgsoft     4163 May  8 13:02 KillAgent.sh
-r-xr-x--- 1 root      cfgsoft     1130 May  8 13:02 killprocs.sh
-r-xr-x--- 1 root      cfgsoft      743 May  8 13:02 stopagent.sh

```

The install directory contains the infrastructure used to install and uninstall the agent. It also contains log files that can be used to determine why the install failed. The `BootStrapInstall.log` file contains a log of all of the actions that the installer took. The `DebugEvent_cis.dbe` is an error log file and can be copied to a collector machine to be viewed using the Debug Event Viewer.

/opt/CMAgent/install:

```
-rw-r----- 1 root      cfgsoft  35332 May  6 16:47 BootStrapInstall.log
-r-xr-x---  1 root      cfgsoft  38940 May  6 16:47 BootStrapInstall.sh
-r--r----- 1 root      cfgsoft    192 Apr 28 10:05 CMAgentPkgReadme.txt
-rw-r----- 1 root      cfgsoft  34376 May  6 16:47 DebugEvent_cis.dbe
-rw-rw----  1 root      cfgsoft    562 May  6 16:46 KillAgent.log
-rw-r----- 1 root      cfgsoft    240 May  5 08:36 checksum
drwxr-x---  2 root      cfgsoft   1432 May  6 16:47 cis
-r--r----- 1 root      cfgsoft   7104 May  6 16:47 csi.config
-rw-r----- 1 root      cfgsoft 365329 May  6 16:47 install.log
-rwxr-x---  1 root      cfgsoft    69 May  5 08:36 package.sizes.Linux
drwxr-x---  3 root      cfgsoft   1048 May  6 16:47 python
-rw-rw----  1 root      cfgsoft  11103 May  6 16:47 reinstall.log
dr-----  2 root      cfgsoft    120 May  6 16:47 saved
-rw-r----- 1 root      cfgsoft    849 May  6 16:47 status
drwxr-x---  2 root      cfgsoft    184 May  6 16:47 uninstall
```

/opt/CMAgent/install/cis:

```
-r--r----- 1 root      cfgsoft   6935 May  5 08:36 CInstallPackage.py
-rw-r----- 1 root      cfgsoft   3321 May  6 16:47 CInstallPackage.pyc
-r--r----- 1 root      cfgsoft   2788 May  5 08:36 CUninstallProduct.py
-rw-r----- 1 root      cfgsoft   1268 May  6 16:47 CUninstallProduct.pyc
```

<--- snip --->

```
-r--r----- 1 root      cfgsoft  48533 May  5 08:36 UserGroup.py
-rw-r----- 1 root      cfgsoft  40251 May  6 16:47 UserGroup.pyc
-r-xr-x---  1 root      cfgsoft 882696 May  5 08:36 _cis.so
-rw-r----- 1 root      cfgsoft    919 May  5 08:36 cis.1.0.Linux
-r--r----- 1 root      cfgsoft    953 May  5 08:36 cis.py
-rw-r----- 1 root      cfgsoft    284 May  6 16:47 cis.pyc
```

/opt/CMAgent/install/python:

```
-r--r----- 1 root      cfgsoft   5610 May  5 08:36 UserDict.py
-rw-r----- 1 root      cfgsoft  11066 May  6 16:47 UserDict.pyc
-r--r----- 1 root      cfgsoft  22409 May  5 08:36 codecs.py
-rw-r----- 1 root      cfgsoft  27697 May  6 16:47 codecs.pyc
-r--r----- 1 root      cfgsoft   6433 May  5 08:36 copy_reg.py
```

```

-rw-r----- 1 root      cfgsoft  6161 May  6 16:47 copy_reg.pyc
<--- snip --->
-r--r----- 1 root      cfgsoft  2244 May  5 08:36 types.py
-rw-r----- 1 root      cfgsoft  3154 May  6 16:47 types.pyc
-r--r----- 1 root      cfgsoft  9092 May  5 08:36 warnings.py
-rw-r----- 1 root      cfgsoft 10055 May  6 16:47 warnings.pyc

/opt/CMAgent/install/python/encodings:
-r--r----- 1 root      cfgsoft  4768 May  5 08:36 __init__.py
-rw-r----- 1 root      cfgsoft  4288 May  6 16:47 __init__.pyc
-r--r----- 1 root      cfgsoft   639 May  5 08:36 utf_8.py

/opt/CMAgent/install/uninstall:
-rw-r--r-- 1 root      cfgsoft 12346 May  6 16:47 Agent.py
-rw-r--r-- 1 root      cfgsoft 11686 May  6 16:47 CFC.py
-rw-r--r-- 1 root      cfgsoft 12804 May  6 16:47 ECMu.py
-rw-r--r-- 1 root      cfgsoft  5182 May  6 16:47 ThirdParty.py
-rw-r--r-- 1 root      cfgsoft    20 May  6 16:47 timestamp.py

```

The ThirdParty sub-directory contains code libraries that are common components. It contains a sub-directory for the code version (1.0) which in turn contains a directory with all of the binaries.

```

/opt/CMAgent/ThirdParty:
dr-xr-x--x 4 root      cfgsoft   96 May  6 16:47 1.0
-rw-r----- 1 root      cfgsoft  594 May  6 16:47 manifest_
ThirdParty.1.0.Linux

/opt/CMAgent/ThirdParty/1.0:
dr-xr-x--x 2 root      cfgsoft  208 May  6 16:47 bin
dr-xr-x--x 2 root      cfgsoft  384 May  6 16:47 lib

```

Notice that the gawk executable is world executable. This allows non privileged inspectors to use it. On a Solaris agent, this directory will also contain the libconv.so.2.1.0 library which is also world readable as it is used by the gawk executable.

```

/opt/CMAgent/ThirdParty/1.0/bin:
-r-xr-x--- 1 root      cfgsoft  4376 May  6 16:47 VMwareFingerPrint
-r-xr-x--- 1 root      cfgsoft 308956 May  6 16:47 gawk

```

```

-r-xr-x--- 1 root    cfgsoft  30736 May  6 16:47 lm
-r-xr-x--- 1 root    cfgsoft   9352 May  6 16:47 lmmgr
-r-xr-x--- 1 root    cfgsoft 104992 May  6 16:47 unzip
-r-xr-x--- 1 root    cfgsoft  61640 May  6 16:47 zip

/opt/CMAgent/ThirdParty/1.0/lib
lrwxrwxrwx 1 root    cfgsoft      47 May  6 16:47 libACE.so ->
/opt/CMAgent/ThirdParty/1.0/lib/libACE.so.5.3.0
-r-xr-x--- 1 root    cfgsoft 1610400 May  6 16:47 libACE.so.5.3.0
-r-xr-x--- 1 root    cfgsoft  417448 May  6 16:47 libboost_regex.so
lrwxrwxrwx 1 root    cfgsoft      45 May  6 16:47 libgcc_s.so ->
/opt/CMAgent/ThirdParty/1.0/lib/libgcc_s.so.1
-r-xr-xr-x 1 root    cfgsoft   33740 May  6 16:47 libgcc_s.so.1
-r-xr-x--- 1 root    cfgsoft  208152 May  6 16:47 librdmm3.so
-r-xr-x--- 1 root    cfgsoft   80036 May  6 16:47 librdmmpsp3.so
lrwxrwxrwx 1 root    cfgsoft      50 May  6 16:47 libstdc++.so ->
/opt/CMAgent/ThirdParty/1.0/lib/libstdc++.so.6.0.0
lrwxrwxrwx 1 root    cfgsoft      50 May  6 16:47 libstdc++.so.6 ->
/opt/CMAgent/ThirdParty/1.0/lib/libstdc++.so.6.0.0
-r-xr-x--- 1 root    cfgsoft  867468 May  6 16:47 libstdc++.so.6.0.0

```

The uninstall directory contains the script which is run to remove the agent.

```

/opt/CMAgent/uninstall:
-rwxrwx--- 1 root    cfgsoft  49490 May  6 16:47 UninstallCMAgent

```

Collector Certificates

In order for a collector to be authorized to talk to an agent, the Collector's certificate (.pem) must be uploaded to the UNIX Agent's /opt/CMAgent/data/db/PDS/CertStore and the file must be readable by the cfgsoft group. The certificate can be located in \Program Files\VMware\VCM\CollectorData

If the Agent was installed from the package located in the collector's "C:\Program Files\VMware\VCM\Installer\Packages" directory, it will already have that collector's certificate.

If the Agent was installed from the a different collector, the certificate file must be copied to the agent (e.g. via ftp in binary mode).

The certificate file can be located in "C:\Program Files\VMware\VCM\Collector Data" and will be name <ECM_Enterprise_Certificate_GUID>.pem

The certificate can be added to the Agent's certificate store using the CSI_ManageCertificateStore utility. Please note that the certificate information is currently maintained in memory while the Agent is running. Therefore, the Agent must be restarted before certificates newly added to the certificate store will be used. Generally, this will not be must of a problem for Agent's running in inetd mode because they will periodically stop themselves; however, if the Agent is running in daemon mode it will have to be manually bounced.

The following commands can be used once the certificate has been copied to the Agent machine:

HPUX

```
CSI_REGISTRY_PATH=/opt/CMAgent SHLIB_
PATH=/opt/CMAgent/CFC/3.0/lib:/opt/CMAgent/ThirdParty/1.0/lib
/opt/CMAgent/CFC/3.0/bin/CSI_ManageCertificateStore -i -f
```

Solaris and Linux

```
CSI_REGISTRY_PATH=/opt/CMAgent LD_LIBRARY_
PATH=/opt/CMAgent/CFC/3.0/lib:/opt/CMAgent/ThirdParty/1.0/lib
/opt/CMAgent/CFC/3.0/bin/CSI_ManageCertificateStore -i -f
```

MAC

```
CSI_REGISTRY_PATH=/opt/CMAgent DYLD_LIBRARY_
PATH=/opt/CMAgent/CFC/3.0/lib:/opt/CMAgent/ThirdParty/1.0/lib
/opt/CMAgent/CFC/3.0/bin/CSI_ManageCertificateStore -i -f
```

AIX

```
CSI_REGISTRY_PATH=/opt/CMAgent
LIBPATH=/opt/CMAgent/CFC/3.0/lib:/opt/CMAgent/ThirdParty/1.0/lib
/opt/CMAgent/CFC/3.0/bin/CSI_ManageCertificateStore -i -f
```

Patch Assessment

5.0 introduce the ability to perform patch assessment inspections. To support this feature additional metadata was added to the IMD tables on the Collector. This metadata is replicated to the Agent in the same manner as all metadata. On the first communication between the Collector and an Agent all metadata is sent in the initial request to the Agent, and the Agent stores this metadata on a per-Collector basis (/opt/CMAgent/data/db/DtmDB/RDM/CollectorName). On subsequent requests from the Collector only modified metadata will be replicated to the Agent. To accomodate the files needed for patch assessment the directory /opt/CMAgent/data/db/DtmDB/RDM/CollectorName/PatchContent has been added. This directory contains .PLS (Lumension patch data) files for each patch to be evaluated. Because metadata is organized on a per-platform basis (as opposed to a platform+architecture basis) .PLS files for various architectures of a specific platform will all be copied/replicated to the Agent for each collector that send requests to the Agent (for example, for a Solaris 10 x86 Agent, .PLS files for Solaris Sparc 8, 9, and 10 would also be sent to the Agent).

Unfortunately, the size of the initial request sent to Agents for which patch assessment data exists in the Collector causes problems. The size of this initial request can cause the ListenerWorkerDaemon and Agent processes to grow to over 500 MB each. To protect the Agent machine from excessively large VCM processes pre-5.0 Agents are set up to shut themselves down when they exceed ~150MB; post-5.0 Agents are allowed to grow to ~380MB.

Directories Created During an Inspection

The Collector specific data model

When a Collector first contacts the Agent, it will replicate its data model to the Agent. The Agent will store this as a Birdstep database as a sub-directory of the DtmDB/RDM directory. The directory will be named with the name of the Collector. Note that is a direct child of the RDM directory.

If this directory is deleted, the next collection will fail. The subsequent collection will re-replicate the data model.

```
opt/CMAgent/data/db/DtmDB/RDM/DVSUP-INSTALL:
-rw-rw---- 1 nobody  cfgsoft      35508 May 20 10:05 327B23C6-0AE4-
428E-0001-7F00F14D0500_data.dat
-rw-rw---- 1 nobody  cfgsoft      19220 May 20 10:05 327B23C6-0AE4-
428E-0001-7F00F14D0500_keys.dat
-rw-rw---- 1 nobody  cfgsoft      36358 May 20 10:02 643C9869-064D-
428E-0001-7F0063CE0800_data.dat
-rw-rw---- 1 nobody  cfgsoft      18765 May 20 10:02 643C9869-064D-
428E-0001-7F0063CE0800_keys.dat
-rw-rw---- 1 nobody  cfgsoft      27124 May 20 10:05 6B8B4567-0AE2-
428E-0001-7F00A5190200_data.dat
-rw-rw---- 1 nobody  cfgsoft      15808 May 20 10:05 6B8B4567-0AE2-
428E-0001-7F00A5190200_keys.dat
-rw-rw---- 1 nobody  cfgsoft        1865 May 20 10:02 DtmDB.dbd
-rw-rw---- 1 nobody  cfgsoft         498 May 20 14:11 DtmDB.taf
-rw-rw---- 1 nobody  cfgsoft     1325056 May 20 10:05 DtmKeys.dbd
-rw-rw---- 1 nobody  cfgsoft     3078144 May 20 10:05 DtmRecord.dbd
```

The Collector specific inspection directory

When an inspection occurs, a sub-directory of /opt/CMAgent/data will be created for the Collector.

```
/opt/CMAgent/data/DVSUP-INSTALL:
drwxrwx--- 2 nobody  cfgsoft      4096 May 20 14:11 Master
drwxrwx--- 2 nobody  cfgsoft      4096 May 20 14:12 Package
```

The Master directory contains the inspected data and is used when performing deltas. There is 1 .mfl file per data class collected. If these files are deleted, the next collection will act as a full collect.

```

/opt/CMAgent/data/DVSUP-INSTALL/Master:
-rw-rw----    1 nobody   cfgsoft      8516 May 20 14:11
UnixAccountGroup.mfl
-rw-rw----    1 nobody   cfgsoft      13892 May 20 14:11
UnixAccountUser.mfl
-rw-rw----    1 nobody   cfgsoft    2869360 May 20 14:11
UnixFileSystem.mfl

```

The Package directory will temporarily contain the results that are sent back to the Collector. This zrp file can be expanded using the command `/opt/CMAgent/ThirdParty/1.0/bin/unzip *zrp`.

This file is deleted when the Agent receives an acknowledgement that the collector has received it. If you wish to capture the file, the following shell command will capture it as soon as it is created:

```
until ls *zrp 2>/dev/null; do sleep 1; done; cp *zrp save.zrp
```

```

/opt/CMAgent/data/DVSUP-INSTALL/Package:
-rw-rw----    1 root      root          10210 May 20 14:11 215D4C5A-AF55-
40EB-BADD-B634B18EF734.zrp

```

The zrp file may contain a debug event log (dbe file) which is returned to the collector and inserted into the SQL database so that they can be viewed at the collector. However, if this is failing the information, can still be retrieved on the agent by capturing the zrp file and extracting the dbe file which can be copied to the collector to be viewed using the Debug Event Viewer.

Saving Executed Scripts and Results

If the SaveTempScriptFiles entry in `/opt/CMAgent/CSIRegistry` is set to true, then copies of the executed gawk scripts, remote command scripts and their output will be stored in the ScriptFiles directory with the format:

File-Type_XXXXXX where XXXXXX is a unique random alpha-numeric string.

The types are:

- script_XXXXXX – a gawk or remote command script
- hds_XXXXXX – the output of a gawk script
- rcmd_XXXXXX – the output of a remote command

```

/opt/CMAgent/data/ScriptFiles:
total 8
drwxrwx---    2 nobody   cfgsoft      4096 May 20 14:08 .

```

```
drwxrwx---    6 root    cfgsoft    4096 May 20 14:07
```

UNIX Agent Error Scenarios

If the installation reports and error

Copy the DebugEvent_cis.dbe file from /opt/CMAgent/install to a collector machine and view it with the Debug Event Viewer.

The /opt/CMAgent/install/BootstrapInstall.log may also provide information on the type of error.

The /opt/CMAgent/install/csi.config file contains the install configuration, e.g. which user to run the agent as etc and is useful information to have.

The Collector cannot ping the Agent

There can be several causes for not being able to ping the collector.

The following steps can be used to determine if the collector can physically contact the agent machine:

1. Try an nslookup of the Agent. If this fails, manually modify the hosts file on the collector to setup a mapping of the Agent machine name to its IP address.
2. Try to manually ping the agent, from a Command Prompt, type ping <agent-machine>.
3. Try to telnet or ssh (e.g. with Putty <http://www.chiark.greenend.org.uk/~sgtatham/putty/>) to the Agent.

The following steps can be used to determine if the agent is alive and listening.

1. Check the file /opt/CMAgent/PDS/CertStore to make sure that the collector's certificate is installed.
2. On the Agent machine check that COLLECTOR *.pem certificate was pushed to CSI_ManageCertificateStore. Use the commands from 1.2 Collector Certificate Store changing -i -f with -l.
3. On the Agent machine, check the file in /var/log/ messages (Linux) or /var/adm/ messages (Solaris) to see if (x)inetd reported any errors when it was reconfigured to enable the agent.
4. Modify the file /opt/CMAgent/ECMu/1.0/scripts/inetd-agent and add -b immediately before -u. Try to contact the agent again and check the file /var/log/messages (Linux) or /var/adm/messages (Solaris) for entries that are reported by CsiAgentListener.
5. Check the file /var/log/secure (Linux) or /var/adm/messages (Solaris) for entries that show the csi-agent process starting. These may only show up if the machine is set up to log these types of message.
6. Check /var/log/CSI/log/DebugEvent_Default.dbe for errors. This file can be copied to the collector box and viewed with the Debug Event Viewer.
7. On a Linux agent machine, type netstat -l | grep csi-agent which should return
tcp 0 0 :csi-agent *: LISTEN
8. On a Solaris agent, type netstat -a | grep csi-agent which should return
.csi-agent *. 0 0 0 0 LISTEN
9. You can use a machine that has nmap installed (usually a linux machine) to see if the port is open to the network, using the command nmap -sT -v -p 26542 <agent-machine> which will return a display like:
Starting nmap V. 3.00 (www.insecure.org/nmap/)
Host <agent-machine> (<ip-address>) appears to be up ... good.

```

Initiating Connect() Scan against <agent-machine> (<ip-address>)
Adding open port 26542/tcp
The Connect() Scan took 0 seconds to scan 1 ports.
Interesting ports on <agent-machine> (<ip-address>)
Port            State           Service
26542/tcp       open            unknown

Nmap run completed - 1 IP address (1 host up) scanned in 0 seconds

```

10. Try to telnet/ssh to the agent machine on the agent port
e.g. telnet <agent-machine> 26542

If you have top installed on the agent machine, you can monitor to see if the listener fires up when the telnet is tried. Start top, then type u. At the Which User (Blank for All): prompt, type the name of the user that the agent is installed as. Type s. At the Delay between updates: prompt, type 1.

As the telnet is executing, you should see the CsiAgentListener process appear and disappear in the top display. The top package is available for Solaris from <http://www.sunfreeware.com>.

Agent fails to return data

Is the Agent starting up?

First, ensure that the Agent processes are actually running.

1. The command `p s -ef | grep CMAgent | grep -v grep` will show all of the Agent processes.
2. If top is available, it can be used to monitor the processes. Start top, then type u. At the Which User (Blank for All): prompt, type the name of the user that the agent is installed as. Type s. At the Delay between updates: prompt, type 1.

You should see `/opt/CMAgent/ECMu/3.0/bin/Agent` and `opt/CMAgent/CFC/1.0/bin/KSrDaemon` when the agent is executing.

`/opt/CMAgent/CFC/3.0/bin/CsiAgentListener` should be started by (x)inetd each time the collector sends a message to the agent.

`/opt/CMAgent/ThirdParty/1.0/bin/lm` will be shown on Linux, AIX, and HP-UX. This is the external Birdstep Lock Manager which is not required on Solaris as it supports an internal lock manager. On Linux, there will always be a lm process running for the (Status Manager) SM database whenever the Agent process is running. While an inspection is in progress, a second lm process will be executing for the collector specific (DataModel) DtmDB.

The command:

```
find /opt/CMAgent/data -name Master | xargs ls -ldt
```

returns a list of all collector directories with the most recently collected directory at the top of the list.

On Red Hat Enterprise 2.1, you will see many instances of these processes. This is because this version of the OS reports all threads within a process and not just the process itself.

On all other platforms, you should see only one Agent and KSrDaemon process.

You may see other additional processes such as gawk or zip depending on when you issue the ps command.

Also follow the instructions in to ensure that the collector can actually communicate with the agent.

Has the Agent been re-installed

If the Agent has been re-installed, the first collection will always fail. The agent will return an error indicating that the Replication Timestamp is out of sync. Performing a second collection will force the collector to re-replicate the data model to the Agent.

Monitoring the Agent operation

If this is the first collection, you will see a directory /opt/CMAgent/data/db/DtmDB/RDM/<collector-name> appear when the agent starts to process the replicated data model.

If this is the first collection, you will see a directory /opt/CMAgent/data/<collector-name> appear when the agent begins to process the inspection request.

On subsequent collections, you will see a directory /opt/CMAgent/data/<collector-name>/<request-id> appear when the agent begins to process the inspection request. The format of the request-id directory name is like a Windows GUID.

The request-id directory contains files downloaded for remote commands and a Results directory which will contain the files that are to be returned to the agent. The Results directory can be monitored in several ways:

- ls -lt - to see each individual data class's hds file being created. The latest inspected data class will appear at the top of the list. If the Agent seems to be hung, this will show which data class it was processing at the time. Performing a ps -ef | grep gawk | grep -v grep will show if the Agent is actively inspecting the data class and may show if the gawk script has hung. An example would be that gawk is trying to perform a checksum on a pipe file which will hang forever.
- ls | wc -l - this will report the number of files in the directory. A collection of all data classes will generally create about 40 files in this directory including hds files and FileUpload* files. It may also contain a Debug Event file.

Once all the inspections have completed, the contents of the Results directory are zipped up and placed as a <request-id>.zrp file in the /opt/CMAgent/data/<collector-name>/Package directory. The Results directory is then deleted. If this never occurs, you can look for a dbf file in the Results directory.

The <request-id>.zrp file is returned to the Collector and deleted once the Collector acknowledges receipt of it. If you need to examine the zrp file on the Agent, it is useful to capture it as soon as it gets created. The shell command until ls *zrp 2>/dev/null; do sleep 1; done; cp *zrp save.zrp is useful for capturing the zrp file as soon as it is created.

Log files to check

The file DebugEvent_Default.dbf located in /var/log/CSI/log (Linux) or /var/adm/CSI/log (Solaris) contains errors reported by the Agent processes which are not specific to a request – for example this file will contain errors indicating that an unauthorized collector tried to contact the agent.

Once the agent begins processing a request, it switches log files and will write to a dbf file in /opt/CSI/data/<collector-name>/<request-id>. This file will eventually be returned to the collector. It can be accessed on the agent as described in

As with the Windows Agent, many of the errors in the dbf files require intimate knowledge of the Agent code in order to decipher them.

The file messages in /var/log (Linux) or /var/adm (Solaris) will show any errors that the Agent processes had before being able to write to the DebugEvent_Default.dbe file. You may need to check the /etc/syslog.conf to validate that errors are being written to this file.

You can also cause the agent to log some informational messages to the messages file. In the file /opt/CMAgent/ECMu/1.0/scripts/inetd-agent, add a -b immediately before the -u. This will cause the CsiAgentListener to log information about which user it is using, the group it is using and each step as it daemonizes itself to detach from (x)inetd. It will also log when it exits. Again, you will need to check the /etc/syslog.conf to validate that infos are being written to the messages file.

If you suspect that there is a race condition in the startup/shutdown and communications between the CsiAgentListener and the Agent or KSrDaemon, you can cause informational messages to be logged to the messages file. These will show each of the processes starting up, shutting down, checking to see if it is safe to shutdown and determining that the process is available to be contacted. In the /opt/CMAgent/ECMu/1.0/scripts/inetd-agent, add LOG_INFOS=1 immediately before CSI_REGISTRY_PATH.

If inspections are failing, you can check /var/log/secure (Linux) or /var/adm/messages (Solaris) for messages:

- CSISecureHigh: Errors from the RunHigh executable meaning that there is some violation of the rules enforced by the suid program for running root privilege inspections.
- CSISecureLow: Errors from the RunLow executable meaning that there is some violation of the rules enforced by the sgid program for running non-root privilege inspections.
- CSISecure: Errors from the RunRemote executable meaning that there is some violation of the rules enforced by the suid program for running root privilege remote commands. This executable is also used in the File Upload job when it needs to copy a file that has restricted permissions.

The CSISecure* messages are deliberately not very useful in debugging as they would allow someone to see the rules being enforced. Initial things to check are:

1. The user configured to run the Agent is a member of the cfgsoft group.
2. The user configured to run the Agent doesn't have a default group of cfgsoft.
3. The user configured to run the Agent has a no-login shell.

These executables cannot be run manually or an error will be logged.

If inspections are not returning all of the expected information, the dbe file in the Collector's request directory can be examined. The file will display any errors that were reported to the stderr of the gawk inspector. These messages will be from the ReadInternal function of the CEcmScriptResultStream class. The message will indicate which data class was being processed at the time. The error may span several sequential messages.

If the Agent does return a zrp file to the collector, any debug events will be logged in the SQL database and can also be observed there. It is also possible to save the .zrp file by enabling the 'SAS: Should SAS data files be kept on the disk after they have been processed' or the 'ETL: Should ETL data files be kept on the disk after they have been processed' settings in the Administration->Settings->General Settings->Database grid of the VCM UI. The .zrp file will be called Results.zrp and will be stored in a request sub-directory of the DSRoot directory.

Monitoring network traffic

In very rare cases, it may be useful to monitor the tcp traffic for the agent. The easiest way to do this is if you have X windows access to the client and the ethereal package is installed.

Ethereal

First, set up ethereal to capture the data:

1. From the ethereal tool bar click on **Capture | Capture Filters**.
2. In the **Filter name** text box, type `csi`.
3. In the **Filter string** text box type `port 26542`.
4. Click **New**.
5. Click **Save**.
6. Click **Close**.

Setup ethereal to display the start of meaningful messages in colors.

1. From the ethereal tool bar click **View | Coloring Rules**.
2. Click **New**.
3. In the **Name** text box, type `Agent Responses`.
4. In the **String** text box, type `data contains HTTP and data contains 200 and data contains OK`.
5. Click **Foreground color**, and then select a color (e.g. blue).
6. Click **OK**.
7. Click **New**.
8. In the **Name** text box, type `VCM Pings`.
9. In the **String** text box type `data contains HTTP and data contains ping`.
10. Click **Foreground color**, and then select a color (e.g. green.)
11. Click **OK**.
12. Click **New**.
13. In the **Name** text box, type `VCM Commands`.
14. In the **String** text box type `data contains HTTP and data contains POST and data contains execute`.
15. Click **Foreground color**, and the select a color (e.g. red).
16. Click **OK**.
17. Click **Save**.
18. Click **OK**.

Start the capture

1. From the ethereal tool bar click on **Capture | Start**.
2. In the **Capture Options**, click **Capture Filter**.
3. Select `csi`.
4. Click **OK**.

As messages flow in and out of the Agent's port (26542), they will appear in the ethereal display. By looking at the colored messages in the display, it is possible to determine what messages have occurred.

For a single inspection, expect to see the following:

- Green – The Ping
- Blue – The ping result – the Content-Length in this message is usually 164 bytes.
- Red – Session Negotiation – the Content-Length in this message is usually 2,222 bytes.
- Blue – Negotiation Complete – the Content-Length in this message is usually 3,538 bytes.
- Red – Inspection Request – the Content-Length varies based on number of data classes selected and if replication is occurring. For example a full collect of Machine.General with full data model replication has a Content-Length of 71,934 bytes
- Blue – Request Scheduled – the Content-Length is usually 8,386 bytes.
- A repeated set of 1 or more
 - Red – Session Negotiation – Content-Length is again 2,222 bytes.
 - Blue – Negotiation Complete – Content-Length is again 3,538 bytes.
 - Red - Check Status – Content-Length is usually 2,618 bytes.
 - Blue – Current Status – Content-Length varies depending on the status of the request. For example, if the request is still in progress the Content-Length is usually 9,110 bytes but when the request is complete, the Content-Length is 8,330 bytes.
- Red – Session Negotiation – Content-Length is again 2,222 bytes.
- Blue – Negotiation Complete – Content-Length is again 3,538 bytes.
- Red – Transfer Results – Content-Length is usually 2,254 bytes.
- Blue – Transferred data – Content-Length is variable based on the number of data classes inspected, delta/full, OS. For example a full collect of Machine.General on a Red Hat 9 platform has a Content-Length of 6,618 bytes.
- Red – Session Negotiation – Content-Length is again 2,222 bytes.
- Blue – Negotiation Complete – Content-Length is again 3,538 bytes.
- Red – Acknowledge Transfer – Content-Length is usually 2,630 bytes.
- Blue – Acknowledged – Content-Length is usually 7,554 bytes.

The message sizes indicated are current as of VCM UNIX/Linux Agent build 1.0.0.1270.

tcpdump

The tcpdump command can be used to gather the same data in a much less user friendly manner.

Start tcpdump using the following command:

```
/usr/sbin/tcpdump -s -l 256 -x -X port 26542 | tee tcpdump.log
```

The tcpdump.log file contains similar information as described for ethereal except that there is no color highlighting, making it necessary to search for the string HTTP to determine where each of the different message types begins.

The Collector reports the job succeeded, but there is still no data

Determine if there are problems with the suid programs in /opt/CMAgent/ECMu/1.0/bin

The permissions should be:

RunHigh: owner root, group cfgsoft, mode r-sr-x---

RunLow: owner csi_acct, group csi_acct, mode r-xr-s---

RunRemote: owner root, group cfgsoft, mode r-sr-x---

NOTE The csi_acct name may be different if the Agent is installed using a different account.

Account and Group Information

The csi_acct must also be properly created. The csi_acct user should not have a shell that permits logins. The shell for csi_acct must be listed in the CSIRegistry's "NoLoginShells", and the no login shell must exist on the system. The primary group for the csi_acct user by default is the csi_acct group (like the csi_acct user name this group name can be changed during Agent install to use another name or an existing group. Using an existing group may cause security risks depending on the privileges that group has); this group is given no elevated permissions (like the standard "nobody" group). The cfgsoft group must always be created; this name must be used. The csi_acct user must be a member of the cfgsoft group, but the cfgsoft group should not be the csi_acct's primary group. If the install creates these groups/account, the uninstall will remove them; if they were pre-existing, the uninstall will not remove them.

If the permissions are correct, then check the db for errors that RunHigh, RunLow, and/or RunRemote failed. In this case, it will not tell you the cause of the error. RunHigh/RunLow will log some generic error messages to syslog on failure; however, for better messages you can rebuild the RunHigh/Low/Remote program with more detailed logging enabled: Search for a commented out syslog entry in the code. Enabling detailed logging gives an error message in syslog containing an error code which can then be located in the source file to determine the error condition. This is deliberately obfuscated so prevent a leak of error information that could enable an 'attack'.

When troubleshooting the setuid binaries, nsswitch.conf should also be checked to confirm that all user lookups are going to the files first. If not, the users may need to be created in YP/LDAP/AD, etc. A common problem is that the user is partially created in the cloud, so the security checks fail. If NONE of the user information is in the cloud, the secondary check to files should work properly.

Also, the mount options for the file system should be checked. A common security practice is to mount /usr, /opt, and /usr/local with 'notsetuid/nosuid' options to prevent setuid binaries from running. This will prevent RunHigh/RunLow/RunRemote from functioning.

Index

A		
about this book	5	
agent	15	
collector	35	
communication protocol	45	
manual installation	44	
network	35	
software provisioning logs	32	
UNIX/Linux	15	
AreResultsSaved	21	
ARS files	21	
authentication	14	
C		
certificates		
UNIX agent	56	
collector		
agent	35	
software provisioning logs	32	
communication protocol		
agent	45	
D		
debug log	10	
diagnostic files		
ARS files	21	
debug	19	
ETL files	23	
event logs	22	
IE tool logs	22	
IIS logs	21	
installation logs	23	
patching debug information	23	
screenshots	19	
SQL Server logs	20	
syslog files	22	
system information	23	
diagram		
workflow	7	
directory structure		
UNIX agent	47	
E		
errors		
package studio	28	
software repositories	25	
UNIX agent	60	
ETL files	23	
event logs	22	
external factors	9	
H		
hardware	17	
I		
IE tool logs	22	
IIS	16	
logs	21	
inspections		
UNIX agent		
directories	58	
installation logs	23	
Internet Information Services	16	
isolate problem	8	
M		
manual installation		
agent	44	
N		
negative behavior	7	
network	17	
agent	35	
P		
package studio	28	
patch assessment	57	
patching debug information	23	
performance	17	
problem		
debug log	10	
external factors	9	
isolate	8	
negative behavior	7	
R		
Report Server	15	
S		
screenshots	19	
scripts		
UNIX agent	59	
software provisioning logs		
agent	32	
collector	32	
software repositories	25	
SQL Server	14	
logs	20	
syslog files	22	
system information	23	

U**UNIX agent**

collector certificates	56
directories	
inspections	58
directory structure	47
errors	60
patch assessment	57
results	59
scripts	59
user interface	13

W**workflow**

diagram	7
---------	---